Core Printing Reference

Graphics & Imaging > Printing

2007-07-24

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

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

Core Printing Reference 11

Core Printing Reference

Framework:	ApplicationServices/ApplicationServices.h
Declared in	PMCore.h PMCoreDeprecated.h PMDefinitions.h PMDefinitionsDeprecated.h

Overview

Core Printing is a C API that Mac OS X applications and command line tools can use to perform printing tasks that don't display a user interface. Core Printing defines a set of opaque types and a rich set of operations on instances of these types. The Core Printing opaque types include:

- `PMPrintSession` for general information about a print job
- `PMPrintSettings` for print job parameters
- `PMPageFormat` for the page format of a printed document
- `PMPaper` for information about a type of paper
- `PMPrinter` for information about a printer

In Carbon applications, Core Printing is used together with Carbon Printing to implement printing features. For more information about Carbon Printing, see *Carbon Printing Reference*.

In Cocoa applications, Core Printing can be used to extend the functionality in the Cocoa printing classes. The `NSPrintInfo` class provides direct access to some Core Printing objects.

> **Note:** Core Printing is available to 64-bit applications, except for functions, data types, and constants that have been deprecated.

Functions by Task

Releasing and Retaining Printing Objects

`PMRelease` (page 76)
Releases a printing object by decrementing its reference count.

`PMRetain` (page 77)
Retains a printing object by incrementing its reference count.

Creating and Using Page Format Objects

PMCreatePageFormat (page 29)
: Creates a new page format object.

PMCreatePageFormatWithPMPaper (page 30)
: Creates a page format object with a specified paper.

PMCopyPageFormat (page 27)
: Copies the settings from one page format object into another.

PMSessionDefaultPageFormat (page 85)
: Assigns default parameter values to a page format object used in the specified printing session.

PMSessionValidatePageFormat (page 93)
: Updates the values in a page format object and validates them against the current formatting printer.

PMSessionCreatePageFormatList (page 83)
: Obtains a list of page format objects, each of which describes a paper size available on the specified printer.

PMPageFormatCreateDataRepresentation (page 41)
: Creates a data representation of a page format object.

PMPageFormatCreateWithDataRepresentation (page 42)
: Creates a page format object from a data representation.

PMFlattenPageFormat (page 171) Deprecated in Mac OS X v10.5
: Flattens a page format object into a Memory Manager handle for storage in a user document. (Deprecated. Use PMPageFormatCreateDataRepresentation (page 41) instead.)

PMFlattenPageFormatToCFData (page 172) Deprecated in Mac OS X v10.5
: Flattens a page format object into a Core Foundation data object for storage in a user document. (Deprecated. Use PMPageFormatCreateDataRepresentation (page 41) instead.)

PMFlattenPageFormatToURL (page 172) Deprecated in Mac OS X v10.5
: Flattens a page format object into a file for storage in a user document. (Deprecated. Use PMPageFormatCreateDataRepresentation (page 41) and write the resulting data to your destination.)

PMUnflattenPageFormat (page 186) Deprecated in Mac OS X v10.5
: Rebuilds a page format object from a Memory Manager handle that contains flattened page format data. (Deprecated. Use PMPageFormatCreateWithDataRepresentation (page 42) instead.)

PMUnflattenPageFormatWithCFData (page 187) Deprecated in Mac OS X v10.5
: Rebuilds a page format object from a Core Foundation data object that contains flattened page format data. (Deprecated. Use PMPageFormatCreateWithDataRepresentation (page 42) instead.)

PMUnflattenPageFormatWithURL (page 188) Deprecated in Mac OS X v10.5
: Rebuilds a page format object from a file system URL that contains flattened page format data. (Deprecated. Instead read the data into a CFData object and use PMPageFormatCreateWithDataRepresentation (page 42).)

Accessing Data in Page Format Objects

PMGetPageFormatExtendedData (page 37)
: Obtains extended page format data previously stored by your application.

`PMSetPageFormatExtendedData` (page 100)
: Stores your application-specific data in a page format object.

`PMGetPageFormatPaper` (page 38)
: Obtains the paper associated with a page format object.

`PMPageFormatGetPrinterID` (page 43)
: Obtains the identifier of the formatting printer for a page format object.

`PMGetOrientation` (page 36)
: Obtains the current setting for page orientation.

`PMSetOrientation` (page 99)
: Sets the page orientation for printing.

`PMGetScale` (page 40)
: Obtains the scaling factor currently applied to the page and paper rectangles.

`PMSetScale` (page 103)
: Sets the scaling factor for the page and paper rectangles.

`PMGetAdjustedPageRect` (page 31)
: Obtains the imageable area or page rectangle, taking into account orientation, application drawing resolution, and scaling settings.

`PMGetAdjustedPaperRect` (page 32)
: Obtains the rectangle defining the paper size, taking into account orientation, application drawing resolution, and scaling settings.

`PMGetUnadjustedPageRect` (page 40)
: Obtains the imageable area or page rectangle, unaffected by orientation, resolution, or scaling.

`PMGetUnadjustedPaperRect` (page 41)
: Obtains the paper rectangle, unaffected by rotation, resolution, or scaling.

`PMGetResolution` (page 176) Deprecated in Mac OS X v10.5
: Obtains the current application's drawing resolution. (Deprecated. Draw using Quartz 2D and call `CGContextScaleCTM` instead.)

`PMSetAdjustedPageRect` (page 182) Deprecated in Mac OS X v10.5
: Requests a particular page size, adjusted for the current rotation, resolution, or scaling settings. (Deprecated. To set a particular paper size and margins, obtain or create a `PMPaper` (page 109) object and call `PMCreatePageFormatWithPMPaper` (page 30).)

`PMSetResolution` (page 185) Deprecated in Mac OS X v10.5
: Sets the application drawing resolution. (Deprecated. Draw using Quartz 2D and call `CGContextScaleCTM` instead.)

`PMSetUnadjustedPaperRect` (page 185) Deprecated in Mac OS X v10.5
: Requests a particular paper size, unaffected by rotation, resolution, or scaling. (Deprecated. To set a particular paper size, obtain or create a `PMPaper` (page 109) object and call `PMCreatePageFormatWithPMPaper` (page 30).)

Creating and Using Print Settings Objects

`PMCreatePrintSettings` (page 30)
: Creates a new print settings object.

`PMSessionDefaultPrintSettings` (page 85)
: Assigns default parameter values to a print settings object for the specified printing session.

PMSessionValidatePrintSettings (page 94)
Validates a print settings object within the context of the specified printing session.

PMPrintSettingsCreateDataRepresentation (page 71)
Creates a data representation of a print settings object.

PMPrintSettingsCreateWithDataRepresentation (page 72)
Creates a print settings object from a data representation.

PMCopyPrintSettings (page 28)
Copies the settings from one print settings object into another.

PMPrintSettingsToOptions (page 75)
Converts print settings into a CUPS options string.

PMPrintSettingsToOptionsWithPrinterAndPageFormat (page 76)
Converts print settings and page format data into a CUPS options string for a specified printer.

PMFlattenPrintSettings (page 173) Deprecated in Mac OS X v10.5
Flattens a print settings object into a Memory Manager handle for storage in a user document. (Deprecated. Use PMPrintSettingsCreateDataRepresentation (page 71) instead.)

PMFlattenPrintSettingsToCFData (page 174) Deprecated in Mac OS X v10.5
Flattens a print settings object into a Core Foundation data object for storage in a user document. (Deprecated. Use PMPrintSettingsCreateDataRepresentation (page 71) instead.)

PMFlattenPrintSettingsToURL (page 174) Deprecated in Mac OS X v10.5
Flattens a print settings object into a URL for storage in a user document. (Deprecated. Instead use PMPrintSettingsCreateDataRepresentation (page 71) and write the resulting data to your destination.)

PMUnflattenPrintSettings (page 188) Deprecated in Mac OS X v10.5
Rebuilds a print settings object from a Memory Manager handle that contains flattened print settings data. (Deprecated. Use PMPrintSettingsCreateWithDataRepresentation (page 72) instead.)

PMUnflattenPrintSettingsWithCFData (page 189) Deprecated in Mac OS X v10.5
Rebuilds a print settings object from a Core Foundation data object that contains flattened print settings data. (Deprecated. Use PMPrintSettingsCreateWithDataRepresentation (page 72) instead.)

PMUnflattenPrintSettingsWithURL (page 190) Deprecated in Mac OS X v10.5
Rebuilds a print settings object from a file that contains flattened print settings data. (Deprecated. Instead read the data into a CFData object and use PMPrintSettingsCreateWithDataRepresentation (page 72).)

Accessing Data in Print Settings Objects

PMGetPrintSettingsExtendedData (page 39)
Obtains extended print settings data previously stored by your application.

PMSetPrintSettingsExtendedData (page 102)
Stores your application-specific data in a print settings object.

PMGetFirstPage (page 34)
Obtains the number of the first page to be printed.

PMSetFirstPage (page 97)
Sets the default page number of the first page to be printed.

PMGetLastPage (page 35)
Obtains the number of the last page to be printed.

PMSetLastPage (page 98)
Sets the page number of the last page to be printed.

PMGetPageRange (page 38)
Obtains the valid range of pages that can be printed.

PMSetPageRange (page 101)
Sets the valid range of pages that can be printed.

PMPrintSettingsGetJobName (page 72)
Obtains the name of a print job.

PMPrintSettingsSetJobName (page 74)
Specifies the name of a print job.

PMGetCopies (page 33)
Obtains the number of copies that the user requests to be printed.

PMSetCopies (page 96)
Sets the initial value for the number of copies to be printed.

PMGetCollate (page 33)
Obtains a Boolean value that indicates whether the job collate option is selected.

PMSetCollate (page 95)
Specifies whether the job collate option is selected.

PMGetDuplex (page 34)
Obtains the selected duplex mode.

PMSetDuplex (page 97)
Sets the duplex mode.

PMPrintSettingsGetValue (page 73)
Obtains the value of a setting in a print settings object.

PMPrintSettingsSetValue (page 74)
Stores the value of a setting in a print settings object.

PMPrintSettingsCopyAsDictionary (page 69)
Creates a dictionary that contains the settings in a print settings object.

PMPrintSettingsCopyKeys (page 70)
Obtains the keys for items in a print settings object.

PMGetJobNameCFString (page 176) Deprecated in Mac OS X v10.5
Obtains the name of the print job. (Deprecated. Use PMPrintSettingsGetJobName (page 72) instead.)

PMSetJobNameCFString (page 183) Deprecated in Mac OS X v10.5
Specifies the name of a print job. (Deprecated. Use PMPrintSettingsSetJobName (page 74) instead.)

Creating Printing Session Objects

PMCreateSession (page 31)
Creates and initializes a printing session object and creates a context for printing operations.

Accessing Data in Printing Session Objects

PMSessionGetDataFromSession (page 89)
Obtains application-specific data previously stored in a printing session object.

PMSessionSetDataInSession (page 91)
Stores your application-specific data in a printing session object.

PMSessionGetCurrentPrinter (page 88)
Obtains the current printer associated with a printing session.

PMSessionSetCurrentPMPrinter (page 91)
Changes the current printer for a printing session.

PMSessionGetCGGraphicsContext (page 88)
Obtains the Quartz graphics context for the current page in a printing session.

PMSessionError (page 87)
Obtains the result code for any error returned by the printing session.

PMSessionSetError (page 93)
Sets the value of the current result code for the specified printing session.

PMSessionGetGraphicsContext (page 181) Deprecated in Mac OS X v10.5
Obtains the graphics context for the current page in a printing session. (Deprecated. Use PMSessionGetCGGraphicsContext (page 88) instead.)

PMSessionGeneral (page 155) Deprecated in Mac OS X v10.4
Maintains compatibility with the PrGeneral function in the classic Printing Manager. (Deprecated. Use PMPrinterGetCommInfo (page 54) instead.)

PMSessionGetDocumentFormatGeneration (page 155) Deprecated in Mac OS X v10.4
Obtains the spool file formats that can be generated for the specified printing session. (Deprecated. If you're drawing using Quartz 2D instead of QuickDraw, use PMSessionBeginCGDocument or PMSessionBeginCGDocumentNoDialog (page 79); for submitting PostScript data, use PMPrinterPrintWithFile (page 65) or PMPrinterPrintWithProvider (page 66); to draw EPS data, use PMCGImageCreateWithEPSDataProvider (page 25).)

PMSessionSetCurrentPrinter (page 162) Deprecated in Mac OS X v10.4
Changes the current printer for a printing session to a printer specified by name. (Deprecated. Use PMSessionSetCurrentPMPrinter (page 91) instead.)

PMSessionSetDocumentFormatGeneration (page 163) Deprecated in Mac OS X v10.4
Requests a specified spool file format and supplies the graphics context type to use for drawing pages within the print loop. (Deprecated. If you're drawing using Quartz 2D instead of QuickDraw, use PMSessionBeginCGDocument or PMSessionBeginCGDocumentNoDialog (page 79); for submitting PostScript data, use PMPrinterPrintWithFile (page 65) or PMPrinterPrintWithProvider (page 66); to draw EPS data, use PMCGImageCreateWithEPSDataProvider (page 25).)

Using Printer Presets

PMPresetCopyName (page 49)
Obtains the localized name for a preset.

PMPresetCreatePrintSettings (page 50)
Creates a print settings object with settings that correspond to a preset.

PMPresetGetAttributes (page 51)
Obtains the attributes of a preset.

Creating and Using Paper Objects

`PMPaperCreateCustom` (page 43)
Creates a custom paper object.

`PMPaperIsCustom` (page 49)
Returns a Boolean value indicating whether a specified paper is a custom paper.

`PMPaperCreate` (page 177) Deprecated in Mac OS X v10.5
Creates a paper object. (Deprecated. Use `PMPrinterGetPaperList` (page 61) to find the built-in papers available for a given printer or use `PMPaperCreateCustom` (page 43) to create a custom paper.)

Accessing Data in Paper Objects

`PMPaperGetID` (page 46)
Obtains the identifier of a paper object.

`PMPaperGetName` (page 47)
Obtains the name for a given paper.

`PMPaperGetWidth` (page 48)
Obtains the width of the sheet of paper represented by a paper object.

`PMPaperGetHeight` (page 45)
Obtains the height of the sheet of paper represented by a paper object.

`PMPaperGetMargins` (page 46)
Obtains the margins describing the unprintable area of the sheet represented by a paper object.

`PMPaperCreateLocalizedName` (page 44)
Obtains the localized name for a given paper.

`PMPaperGetPrinterID` (page 48)
Obtains the printer ID of the printer to which a given paper corresponds.

`PMPaperGetPPDPaperName` (page 47)
Obtains the PPD paper name for a given paper.

Print Loop Functions

`PMSessionBeginCGDocumentNoDialog` (page 79)
Begins a print job that draws into a Quartz graphics context and suppresses the printing status dialog.

`PMSessionEndDocumentNoDialog` (page 86)
Ends a print job started by calling the function `PMSessionBeginCGDocumentNoDialog` (page 79) or `PMSessionBeginDocumentNoDialog` (page 179).

`PMSessionBeginPageNoDialog` (page 80)
Starts a new page for printing in the specified printing session and suppresses the printing status dialog.

`PMSessionEndPageNoDialog` (page 86)
Indicates the end of drawing the current page for the specified printing session.

`PMSessionBeginDocumentNoDialog` (page 179) Deprecated in Mac OS X v10.5
Begins a print job that, by default, draws into a QuickDraw graphics port, and suppresses the printing status dialog. (Deprecated. Use `PMSessionBeginCGDocumentNoDialog` (page 79) instead.)

`PMSessionSetIdleProc` (page 165) Deprecated in Mac OS X v10.4
Installs an idle callback function in your print loop. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

Accessing the Print Job Destination

`PMSessionSetDestination` (page 92)
Sets the destination location, format, and type for a print job.

`PMSessionGetDestinationType` (page 90)
Obtains the output destination for a print job.

`PMSessionCopyDestinationFormat` (page 81)
Obtains the destination format for a print job.

`PMSessionCopyDestinationLocation` (page 82)
Obtains a destination location for a print job.

`PMSessionCopyOutputFormatList` (page 82)
Obtains an array of destination formats supported by the current print destination.

Creating Printer Objects

`PMServerLaunchPrinterBrowser` (page 78)
Launches the printer browser to browse the printers available for a print server.

`PMServerCreatePrinterList` (page 78)
Creates a list of printers available to a print server.

`PMSessionCreatePrinterList` (page 84)
Creates a list of printers available in the specified printing session.

`PMPrinterCreateFromPrinterID` (page 54)
Creates a printer object from a print queue identifier.

`PMCreateGenericPrinter` (page 28)
Creates a generic printer object.

Accessing Information About a Printer

`PMPrinterCopyDescriptionURL` (page 51)
Obtains the URL of the description file for a given printer.

`PMPrinterCopyDeviceURI` (page 52)
Obtains the device URI of a given printer.

`PMPrinterCopyHostName` (page 52)
Obtains the name of the server hosting the print queue for a given printer.

`PMPrinterCopyPresets` (page 53)
Obtains a list of print settings presets for a printer.

`PMPrinterGetCommInfo` (page 54)
Obtains information about the communication channel for a printer.

`PMPrinterGetDriverCreator` (page 55)
Obtains the creator of the driver associated with the specified printer.

`PMPrinterGetDriverReleaseInfo` (page 55)
Obtains version information for the driver associated with the specified printer.

`PMPrinterGetID` (page 56)
Returns the unique identifier of a printer.

`PMPrinterGetLanguageInfo` (page 57)
Obtains information about the imaging language for the specified printer.

`PMPrinterGetLocation` (page 58)
Returns the location of a printer.

`PMPrinterGetMakeAndModelName` (page 58)
Obtains the manufacturer and model name of the specified printer.

`PMPrinterGetMimeTypes` (page 59)
Obtains a list of MIME content types supported by a printer using the specified print settings.

`PMPrinterGetName` (page 60)
Returns the human-readable name of a printer.

`PMPrinterGetOutputResolution` (page 60)
Obtains the printer hardware output resolution for the specified print settings.

`PMPrinterSetOutputResolution` (page 68)
Sets the print settings to reflect the specified printer hardware output resolution.

`PMPrinterGetPaperList` (page 61)
Obtains the list of papers available for a printer.

`PMPrinterGetPrinterResolutionCount` (page 61)
Obtains the number of resolution settings supported by the specified printer.

`PMPrinterGetIndexedPrinterResolution` (page 57)
Obtains a resolution setting based on an index into the range of settings supported by the specified printer.

`PMPrinterGetState` (page 62)
Obtains the current state of the print queue for a printer.

`PMPrinterSetDefault` (page 67)
Sets the default printer for the current user.

`PMPrinterIsDefault` (page 63)
Returns a Boolean value indicating whether a printer is the default printer for the current user.

`PMPrinterIsFavorite` (page 63)
Returns a Boolean value indicating whether a printer is in the user's list of favorite printers.

`PMPrinterIsPostScriptCapable` (page 64)
Returns a Boolean value indicating whether a printer is PostScript capable.

`PMPrinterIsPostScriptPrinter` (page 64)
Determines whether a printer is a PostScript printer.

`PMPrinterIsRemote` (page 65)
Indicates whether a printer is hosted by a remote print server.

PMPrinterGetPrinterResolution (page 178) Deprecated in Mac OS X v10.5
Obtains a resolution setting for the specified printer. (Deprecated. Use PMPrinterGetPrinterResolutionCount (page 61) and PMPrinterGetIndexedPrinterResolution (page 57) to examine the available printer resolutions.)

PMPrinterGetDescriptionURL (page 152) Deprecated in Mac OS X v10.4
Obtains a reference to the specified printer's description file. (Deprecated. Use PMPrinterCopyDescriptionURL (page 51) instead.)

PMPrinterGetDeviceURI (page 153) Deprecated in Mac OS X v10.4
Obtains a copy of a printer's device URI. (Deprecated. Use PMPrinterCopyDeviceURI (page 52) instead.)

Submitting a Print Job to a Printer

PMPrinterPrintWithFile (page 65)
Submits a print job to a specified printer using a file that contains print data.

PMPrinterPrintWithProvider (page 66)
Submits a print job to a specified printer using a Quartz data provider to obtain the print data.

Accessing PostScript Printer Description Files

PMCopyAvailablePPDs (page 26)
Obtains the list of PostScript printer description (PPD) files in a PPD domain.

PMCopyLocalizedPPD (page 26)
Obtains a localized PostScript printer description (PPD) file.

PMCopyPPDData (page 27)
Obtains the uncompressed PPD data for a PostScript printer description (PPD) file.

Printing with PostScript Data

PMCGImageCreateWithEPSDataProvider (page 25)
Creates an image that references both the PostScript contents of EPS data and a preview (proxy) image for the data.

PMPrinterWritePostScriptToURL (page 68)
Converts an input file of the specified MIME type to printer-ready PostScript for a destination printer.

PMSessionPostScriptBegin (page 159) Deprecated in Mac OS X v10.4
Puts the current printer driver into PostScript mode, ready to accept PostScript data instead of QuickDraw data. (Deprecated. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

PMSessionPostScriptData (page 159) Deprecated in Mac OS X v10.4
Passes PostScript data, referenced by a pointer, to the current printer driver. (Deprecated. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

`PMSessionPostScriptEnd` (page 160) Deprecated in Mac OS X v10.4
Restores the current driver to QuickDraw mode, ready to accept QuickDraw data instead of PostScript data. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSessionPostScriptFile` (page 161) Deprecated in Mac OS X v10.4
Passes the PostScript data, contained in a file, to the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSessionPostScriptHandle` (page 162) Deprecated in Mac OS X v10.4
Passes the PostScript data, referenced by a Memory Manager handle, to the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSessionSetPSInjectionData` (page 166) Deprecated in Mac OS X v10.4
Specifies a set of PostScript code injection points and the PostScript data to be injected. (Deprecated. Use `PMPrinterPrintWithFile` (page 65) or `PMPrinterPrintWithProvider` (page 66) instead.)

Using PDF Workflow Items

`PMWorkflowCopyItems` (page 103)
Obtains an array of the available PDF workflow items.

`PMWorkflowSubmitPDFWithOptions` (page 104)
Submits a PDF file for workflow processing using the specified CUPS options string.

`PMWorkflowSubmitPDFWithSettings` (page 105)
Submits a PDF file for workflow processing using the specified print settings.

Matching Color With ColorSync

`PMSessionDisableColorSync` (page 180) Deprecated in Mac OS X v10.5
Disables use of a custom ColorSync profile previously enabled by the function `PMSessionEnableColorSync` (page 181). (Deprecated. There is no replacement; draw using Quartz 2D instead.)

`PMSessionEnableColorSync` (page 181) Deprecated in Mac OS X v10.5
Enables use of a custom ColorSync profile previously set by the function `PMSetProfile` (page 184). (Deprecated. There is no replacement; draw using Quartz 2D instead.)

`PMSetProfile` (page 184) Deprecated in Mac OS X v10.5
Embeds a color profile during printing. (Deprecated. There is no replacement; draw using Quartz 2D instead.)

Converting and Saving Old Print Records

`PMSessionConvertOldPrintRecord` (page 154) Deprecated in Mac OS X v10.4
Creates new page format and print settings objects from an old-style print record created for the classic Printing Manager. (Deprecated. There is no replacement; during the transition from Mac OS 9 to Mac OS X, this function facilitated the migration of print records saved in documents created in Mac OS 9, but the function no longer serves any useful purpose in Mac OS X.)

`PMSessionMakeOldPrintRecord` (page 158) Deprecated in Mac OS X v10.4
: Creates an old-style print record from page format and print settings objects. (Deprecated. There is no replacement; old-style print records are obsolete and serve no useful purpose in Mac OS X.)

Creating, Calling, and Deleting Universal Procedure Pointers

`DisposePMIdleUPP` (page 135) Deprecated in Mac OS X v10.4
: Disposes of a universal procedure pointer (UPP) to an idle callback. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

`InvokePMIdleUPP` (page 135) Deprecated in Mac OS X v10.4
: Calls an idle callback. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

`NewPMIdleUPP` (page 136) Deprecated in Mac OS X v10.4
: Creates a new universal procedure pointer (UPP) to an idle callback. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

Legacy Core Printing Functions

`PMGetDestination` (page 175) Deprecated in Mac OS X v10.5
: Obtains the output destination of a print job. (Deprecated. Use `PMSessionGetDestinationType` (page 90), `PMSessionCopyDestinationFormat` (page 81), or `PMSessionCopyDestinationLocation` (page 82) instead.)

`PMBegin` (page 136) Deprecated in Mac OS X v10.4
: Prepares Core Printing for use. (Deprecated. Use `PMCreateSession` (page 31) instead.)

`PMConvertOldPrintRecord` (page 136) Deprecated in Mac OS X v10.4
: Creates a new `PMPageFormat` object and a new `PMPrintSettings` object from a print record created by the classic Printing Manager. (Deprecated. There is no replacement; during the transition from Mac OS 9 to Mac OS X, this function facilitated the migration of print records saved in documents created in Mac OS 9, but the function no longer serves any useful purpose in Mac OS X.)

`PMDefaultPageFormat` (page 137) Deprecated in Mac OS X v10.4
: Assigns default parameter values to an existing `PMPageFormat` object, for the current printer. (Deprecated. Use `PMSessionDefaultPageFormat` (page 85) instead.)

`PMDefaultPrintSettings` (page 138) Deprecated in Mac OS X v10.4
: Assigns default parameter values to a `PMPrintSettings` object. (Deprecated. Use `PMSessionDefaultPrintSettings` (page 85) instead.)

`PMDisableColorSync` (page 138) Deprecated in Mac OS X v10.4
: Disables ColorSync color matching for the current page. (Deprecated. There is no replacement; draw using Quartz 2D instead.)

`PMDisposePageFormat` (page 138) Deprecated in Mac OS X v10.4
: Releases memory previously allocated for a `PMPageFormat` object. (Deprecated. Use `PMRelease` (page 76) instead.)

`PMDisposePrintSettings` (page 139) Deprecated in Mac OS X v10.4
: Releases memory previously allocated for a `PMPrintSettings` object. (Deprecated. Use `PMRelease` (page 76) instead.)

`PMEnableColorSync` (page 139) Deprecated in Mac OS X v10.4
: Enables ColorSync color matching for the current page. (Deprecated. There is no replacement; draw using Quartz 2D instead.)

`PMEnd` (page 140) Deprecated in Mac OS X v10.4
: Closes Core Printing and releases its allocated memory. (Deprecated. Use `PMRelease` (page 76) to release a `PMPrintSession` (page 110) object instead.)

`PMError` (page 140) Deprecated in Mac OS X v10.4
: Obtains the result code from the last printing function called by your application. (Deprecated. Use `PMSessionError` (page 87) instead.)

`PMGeneral` (page 141) Deprecated in Mac OS X v10.4
: Maintains compatibility with the `PrGeneral` function in the classic Printing Manager. (Deprecated. Use `PMPrinterGetCommInfo` (page 54) instead.)

`PMGetColorMode` (page 141) Deprecated in Mac OS X v10.4
: Obtains the color mode for the print job. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

`PMGetDriverCreator` (page 142) Deprecated in Mac OS X v10.4
: Obtains the creator of the driver associated with the current printer. (Deprecated. Use `PMPrinterGetDriverCreator` (page 55) instead.)

`PMGetDriverReleaseInfo` (page 142) Deprecated in Mac OS X v10.4
: Obtains release information for the driver associated with the current printer. (Deprecated. Use `PMPrinterGetDriverReleaseInfo` (page 55) instead.)

`PMGetGrafPtr` (page 143) Deprecated in Mac OS X v10.4
: Obtains the printing port from an opaque printing context. (Deprecated. Use `PMSessionGetCGGraphicsContext` (page 88) instead.)

`PMGetIndexedPrinterResolution` (page 143) Deprecated in Mac OS X v10.4
: Obtains a resolution setting based on an index into the range of settings supported by the current printer. (Deprecated. Use `PMPrinterGetIndexedPrinterResolution` (page 57) instead.)

`PMGetJobName` (page 144) Deprecated in Mac OS X v10.4
: Obtains the name of the print job. (Deprecated. Use `PMPrintSettingsGetJobName` (page 72) instead.)

`PMGetLanguageInfo` (page 144) Deprecated in Mac OS X v10.4
: Obtains information about the current printer's imaging language. (Deprecated. Use `PMPrinterGetLanguageInfo` (page 57) instead.)

`PMGetPhysicalPageSize` (page 145) Deprecated in Mac OS X v10.4
: Obtains the size of the imageable area in points, unaffected by rotation, resolution, or scaling. (Deprecated. Use `PMGetUnadjustedPageRect` (page 40) or examine the paper returned by `PMGetPageFormatPaper` (page 38).)

`PMGetPhysicalPaperSize` (page 146) Deprecated in Mac OS X v10.4
: Obtains the size of the paper in points, unaffected by rotation, resolution, or scaling. (Deprecated. Use `PMGetUnadjustedPaperRect` (page 41) or examine the paper returned by `PMGetPageFormatPaper` (page 38).)

`PMGetPrinterResolution` (page 146) Deprecated in Mac OS X v10.4
: Obtains the resolution setting for the current printer according to the tag parameter. (Deprecated. Use `PMPrinterGetPrinterResolutionCount` (page 61) and `PMPrinterGetIndexedPrinterResolution` (page 57) to examine the available printer resolutions.)

`PMGetPrinterResolutionCount` (page 147) Deprecated in Mac OS X v10.4
: Obtains the number of resolution settings supported by the current printer. (Deprecated. Use `PMPrinterGetPrinterResolutionCount` (page 61) instead.)

`PMIsPostScriptDriver` (page 148) Deprecated in Mac OS X v10.4
: Reports whether the current printer driver supports the PostScript language. (Deprecated. Use `PMPrinterIsPostScriptCapable` (page 64) or `PMPrinterIsPostScriptPrinter` (page 64) instead.)

`PMMakeOldPrintRecord` (page 148) Deprecated in Mac OS X v10.4
: Creates an old-style print record from a `PMPageFormat` and a `PMPrintSettings` object. (Deprecated. There is no replacement; old-style print records are obsolete and serve no useful purpose in Mac OS X.)

`PMNewPageFormat` (page 149) Deprecated in Mac OS X v10.4
: Creates a new `PMPageFormat` object. (Deprecated. Use `PMCreatePageFormat` (page 29) instead.)

`PMNewPrintSettings` (page 149) Deprecated in Mac OS X v10.4
: Creates a new `PMPrintSettings` object. (Deprecated. Use `PMCreatePrintSettings` (page 30) instead.)

`PMPostScriptBegin` (page 150) Deprecated in Mac OS X v10.4
: Puts the current driver into PostScript mode, ready to accept PostScript data instead of QuickDraw data. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMPostScriptData` (page 150) Deprecated in Mac OS X v10.4
: Passes PostScript data, referenced by a pointer, to the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMPostScriptEnd` (page 151) Deprecated in Mac OS X v10.4
: Restores the current driver to QuickDraw mode, ready to accept QuickDraw data instead of PostScript data. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMPostScriptFile` (page 151) Deprecated in Mac OS X v10.4
: Passes PostScript data, contained in a file, to the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMPostScriptHandle` (page 152) Deprecated in Mac OS X v10.4
: Passes PostScript data, referenced by a handle, to the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSessionGetDocumentFormatSupported` (page 156) Deprecated in Mac OS X v10.4
: Obtains the spool file formats that are accepted by the current printer driver. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSessionIsDocumentFormatSupported` (page 157) Deprecated in Mac OS X v10.4
: Reports whether the current printer driver supports a specified spool file format. (Deprecated. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

`PMSetColorMode` (page 167) Deprecated in Mac OS X v10.4
: Sets the desired color mode for the print job. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

`PMSetError` (page 168) Deprecated in Mac OS X v10.4
: Sets the value of the current result code. (Deprecated. Use `PMSessionSetError` (page 93) instead.)

`PMSetIdleProc` (page 168) Deprecated in Mac OS X v10.4
: Installs an idle callback function in your print loop. (Deprecated. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

`PMSetJobName` (page 169) Deprecated in Mac OS X v10.4
: Specifies the name of the print job. (Deprecated. Use `PMPrintSettingsSetJobName` (page 74) instead.)

`PMSetPhysicalPaperSize` (page 169) Deprecated in Mac OS X v10.4
: Requests a particular paper size, unaffected by rotation, resolution, or scaling. (Deprecated. Use `PMCreatePageFormatWithPMPaper` (page 30) instead.)

`PMValidatePageFormat` (page 170) Deprecated in Mac OS X v10.4
: Obtains a valid `PMPageFormat` object. (Deprecated. Use `PMSessionValidatePageFormat` (page 93) instead.)

`PMValidatePrintSettings` (page 171) Deprecated in Mac OS X v10.4
: Obtains a valid `PMPrintSettings` object. (Deprecated. Use `PMSessionValidatePrintSettings` (page 94) instead.)

Functions

PMCGImageCreateWithEPSDataProvider

Creates an image that references both the PostScript contents of EPS data and a preview (proxy) image for the data.

```
CGImageRef PMCGImageCreateWithEPSDataProvider (
   CGDataProviderRef epsDataProvider,
   CGImageRef epsPreview
);
```

Parameters

epsDataProvider
: A Quartz data provider that supplies the PostScript contents of the EPS file. The EPS data must begin with the EPSF required header and bounding box DSC (Document Structuring Conventions) comments.

epsPreview
: A Quartz image that serves as the proxy image for the EPS file. When the image returned by this function is rendered onscreen or sent to a printer that cannot render PostScript, this proxy image is drawn instead.

Return Value

An image capable of rendering either the EPS content or the proxy image, depending upon the capabilities of the destination printer.

Discussion

It is likely that data will not be read from the EPS data provider until after this function returns. You should be careful not to free the underlying EPS data until the data provider's release function is invoked. Similarly, do not free the preview image data until the image data provider's release function is invoked. You are responsible for releasing the data providers for the EPS image and the EPS preview image.

Note that in Mac OS X v10.3 and later, Quartz can convert EPS data into PDF data. Using this feature and then using Quartz to draw the resulting PDF data may produce superior results for your application. See *CGPSConverter Reference* for details.

Availability

Available in Mac OS X v10.1 and later.

Declared In

`PMCore.h`

PMCopyAvailablePPDs

Obtains the list of PostScript printer description (PPD) files in a PPD domain.

```
OSStatus PMCopyAvailablePPDs (
   PMPPDDomain domain,
   CFArrayRef *ppds
);
```

Parameters

domain

The PPD domain to search. See "PostScript Printer Description File Domains" (page 125) for a description of the constants you can use to specify the domain.

ppds

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array of PPD files in the specified domain. Each element in the array is a Core Foundation URL object that specifies the location of a PPD file or a compressed PPD file. You are responsible for releasing the array. If the specified domain is not valid, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMCopyLocalizedPPD

Obtains a localized PostScript printer description (PPD) file.

```
OSStatus PMCopyLocalizedPPD (
   CFURLRef ppd,
   CFURLRef *localizedPPD
);
```

Parameters

ppd

A Core Foundation URL object for a PPD file. You can obtain a PPD URL using the function `PMCopyAvailablePPDs` (page 26).

localizedPPD

A pointer to your `CFURLRef` variable. On return, the variable refers to a Core Foundation URL object. The URL specifies the location of a PPD file or a compressed PPD file that has been localized for the current user's language preference. You are responsible for releasing the URL. If the *ppd* parameter is not valid, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

To access the data in the PPD file, you can use the function `PMCopyPPDData` (page 27).

Special Considerations

In Mac OS X v10.5 and later, the printing system supports globalized PPD files as defined in CUPS version 1.2 and later. A globalized PPD file contains multiple localizations within a single file. If a globalized PPD file exists, this function returns the URL to this file and it is up to the application to obtain the correct localized data. For more information, see CUPS PPD Extensions.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMCopyPageFormat

Copies the settings from one page format object into another.

```
OSStatus PMCopyPageFormat (
   PMPageFormat formatSrc,
   PMPageFormat formatDest
);
```

Parameters

formatSrc

The page format object to duplicate.

formatDest

The page format object to receive the copied settings. On return, this object contains the same settings as the `formatSrc` object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMCore.h`

PMCopyPPDData

Obtains the uncompressed PPD data for a PostScript printer description (PPD) file.

```
OSStatus PMCopyPPDData (
   CFURLRef ppd,
   CFDataRef *data
);
```

Parameters

ppd

A URL for a PPD or compressed PPD file. You can obtain a PPD URL using the function `PMCopyAvailablePPDs` (page 26) or `PMCopyLocalizedPPD` (page 26).

data

A pointer to your `CFDataRef` variable. On return, the variable refers to a Core Foundation data object containing the uncompressed PPD data from the specified PPD file. You are responsible for releasing the data object. If the *ppd* parameter does not reference a PPD file, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMCopyPrintSettings

Copies the settings from one print settings object into another.

```
OSStatus PMCopyPrintSettings (
   PMPrintSettings settingSrc,
   PMPrintSettings settingDest
);
```

Parameters

settingSrc

The print settings object to duplicate.

settingDest

The print settings object to receive the copied settings. On return, this object contains the same settings as the `settingSrc` object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMCore.h`

PMCreateGenericPrinter

Creates a generic printer object.

```
OSStatus PMCreateGenericPrinter (
    PMPrinter *printer
);
```

Parameters

printer

A pointer to your PMPrinter (page 109) variable. On return, the variable refers to a new printer object that represents the generic formatting printer. You are responsible for releasing the printer object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function provides a way to create a PMPrinter object that represents the generic formatting printer.

Availability

Available in Mac OS X v10.5 and later.

Declared In

PMCore.h

PMCreatePageFormat

Creates a new page format object.

```
OSStatus PMCreatePageFormat (
   PMPageFormat *pageFormat
);
```

Parameters

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a new page format object. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function allocates memory for a new page format object in your application's memory space and sets its reference count to 1. The new page format object is empty and unusable until you call PMSessionDefaultPageFormat (page 85) or PMCopyPageFormat (page 27).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

PMCore.h

PMCreatePageFormatWithPMPaper

Creates a page format object with a specified paper.

```
OSStatus PMCreatePageFormatWithPMPaper (
   PMPageFormat *pageFormat,
   PMPaper paper
);
```

Parameters

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a new page format object that represents the specified paper. You are responsible for releasing the page format object with the function PMRelease (page 76).

paper

The type of paper for the new page format object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

PMCore.h

PMCreatePrintSettings

Creates a new print settings object.

```
OSStatus PMCreatePrintSettings (
   PMPrintSettings *printSettings
);
```

Parameters

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a new print settings object. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function allocates memory for a new print settings object in your application's memory space and sets its reference count to 1. The new print settings object is empty and unusable until you call PMSessionDefaultPrintSettings (page 85) or PMCopyPrintSettings (page 28).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMCreateSession

Creates and initializes a printing session object and creates a context for printing operations.

```
OSStatus PMCreateSession (
   PMPrintSession *printSession
);
```

Parameters

printSession

A pointer to your `PMPrintSession` (page 110) variable. On return, the variable refers to a new printing session object. You are responsible for releasing the printing session object with the function `PMRelease` (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function allocates memory for a new printing session object in your application's memory space and sets its reference count to 1. The new printing session object is initialized with information that the printing system uses for a print job.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMGetAdjustedPageRect

Obtains the imageable area or page rectangle, taking into account orientation, application drawing resolution, and scaling settings.

```
OSStatus PMGetAdjustedPageRect (
   PMPageFormat pageFormat,
   PMRect *pageRect
);
```

Parameters

pageFormat

The page format object whose adjusted page rectangle you want to obtain.

pageRect

A pointer to your `PMRect` (page 111) structure. On return, the structure contains the current imageable area, in points, taking into account scaling, rotation, and application resolution settings. The page rectangle is the area of the page to which an application can draw. The coordinates for the upper-left corner of the page rectangle are (0,0). See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Before using this function, you must call `PMSessionValidatePageFormat` (page 93) to ensure that the values for the adjusted page rectangle correctly account for scaling, rotation, and application resolution settings.

Availability
Available in Mac OS X v10.0 and later.

Related Sample Code
CarbonSketch

Declared In
`PMCore.h`

PMGetAdjustedPaperRect

Obtains the rectangle defining the paper size, taking into account orientation, application drawing resolution, and scaling settings.

```
OSStatus PMGetAdjustedPaperRect (
   PMPageFormat pageFormat,
   PMRect *paperRect
);
```

Parameters

`pageFormat`
: The page format object whose adjusted paper rectangle you want to obtain.

`paperRect`
: A pointer to your `PMRect` (page 111) structure. On return, the structure describes the current paper size, in points, taking into account scaling, rotation, and application resolution settings. The coordinates of the upper-left corner of the paper rectangle are specified relative to the page rectangle. The coordinates of the upper-left corner of the page rectangle are always (0,0), which means the coordinates of the upper-left corner of the paper rectangle are always negative or (0,0). See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Before using this function, you must call the function `PMSessionValidatePageFormat` (page 93) to ensure that the values for the adjusted paper rectangle correctly account for scaling, rotation, and application resolution settings.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMCore.h`

PMGetCollate

Obtains a Boolean value that indicates whether the job collate option is selected.

```
OSStatus PMGetCollate (
   PMPrintSettings printSettings,
   Boolean *collate
);
```

Parameters

printSettings

The print settings object you're querying to determine whether the job collate option is selected.

collate

A pointer to your Boolean variable. On return, `true` if the job collate option is selected; otherwise, `false`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The Collated checkbox is displayed in the Copies & Pages pane of the Print dialog. This option determines how printed material is organized. For example, if you have a document that is three pages long and you are printing multiple copies with the Collated option selected, the job prints pages 1, 2, and 3 in that order and then repeats. However, if the Collated option is not selected and you're printing multiple copies of those same three pages, the job prints copies of page 1, then copies of page 2, and finally copies of page 3.

Availability

Available in Mac OS X v10.2 and later.

See Also

`PMSetCollate` (page 95)

Declared In

`PMCore.h`

PMGetCopies

Obtains the number of copies that the user requests to be printed.

```
OSStatus PMGetCopies (
   PMPrintSettings printSettings,
   UInt32 *copies
);
```

Parameters

printSettings

The print settings object whose number of copies you want to obtain.

copies

A pointer to your `UInt32` variable. On return, the variable contains the number of copies requested by the user.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMSetCopies` (page 96)

Declared In
`PMCore.h`

PMGetDuplex

Obtains the selected duplex mode.

```
OSStatus PMGetDuplex (
   PMPrintSettings printSettings,
   PMDuplexMode *duplexSetting
);
```

Parameters

printSettings

The print settings object whose duplex mode you want to obtain.

duplexSetting

A pointer to your `PMDuplexMode` variable. On return, the variable contains the duplex mode setting in the current print job. Possible values include:

- `kPMDuplexNone` (one-sided printing)
- `kPMDuplexNoTumble` (two-sided printing)
- `kPMDuplexTumble` (two-sided printing with tumbling)

See "Duplex Modes" (page 115) for a full description of the duplex mode constants.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Duplex printing is a print job that prints on both sides of the paper. The Two-Sided printing control is displayed in the Layout pane of the Print dialog.

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

Declared In
`PMCore.h`

PMGetFirstPage

Obtains the number of the first page to be printed.

```
OSStatus PMGetFirstPage (
   PMPrintSettings printSettings,
   UInt32 *first
);
```

Parameters

printSettings

The print settings object whose first page number you want to obtain.

first

A pointer to your UInt32 variable. On return, the variable contains the page number of the first page to print. The default first page number is 1.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can use this function to obtain the page number entered by the user in the From field of the Print dialog. If the user selects the All button, the function returns a value of 1. If the user did not enter a value, the function returns the value of the previous call to PMSetFirstPage, if any, or the default value of 1.

Availability

Available in Mac OS X v10.0 and later.

See Also

PMSetFirstPage (page 97)

Related Sample Code

CarbonSketch

Declared In

PMCore.h

PMGetLastPage

Obtains the number of the last page to be printed.

```
OSStatus PMGetLastPage (
   PMPrintSettings printSettings,
   UInt32 *last
);
```

Parameters

printSettings

The print settings object whose last page number you want to obtain.

last

A pointer to your UInt32 variable. On return, the variable contains the page number of the last page to print.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion
You use this function to obtain the page number entered by the user in the To field of the Print dialog. If the user did not enter a value, the function returns the value of the previous call to `PMSetLastPage`, if any, or a default value.

You should not look for the constant `kPMPrintAllPages`. That constant is used only with the `PMSetLastPage` and `PMSetPageRange` functions to specify a last page. It is not returned by the `PMGetLastPage` function.

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMSetLastPage` (page 98)

Related Sample Code
CarbonSketch

Declared In
`PMCore.h`

PMGetOrientation

Obtains the current setting for page orientation.

```
OSStatus PMGetOrientation (
   PMPageFormat pageFormat,
   PMOrientation *orientation
);
```

Parameters

pageFormat

The page format object whose orientation you want to obtain.

orientation

A pointer to your `PMOrientation` variable. On return, the variable contains a constant value indicating the page orientation. Supported values are:

- `kPMPortrait`
- `kPMLandscape`
- `kPMReversePortrait` (supported in Mac OS X v10.5 and later)
- `kPMReverseLandscape`

See "Page Orientation Constants" (page 117) for a complete description of the page orientation constants.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMSetOrientation` (page 99)

Declared In
`PMCore.h`

PMGetPageFormatExtendedData

Obtains extended page format data previously stored by your application.

```
OSStatus PMGetPageFormatExtendedData (
   PMPageFormat pageFormat,
   OSType dataID,
   UInt32 *size,
   void *extendedData
);
```

Parameters

pageFormat
The page format object that contains your extended data.

dataID
A 4-character code that identifies your data. This is typically your application's creator code. If your creator code is outside the ASCII 7-bit character range 0x20–0x7F, you need to use a different 4-character code.

size
A pointer to a value that specifies the size of the buffer you have allocated for the extended page format data. On return, this variable contains the number of bytes read into the buffer or the size of the extended data. You can pass the constant `kPMDontWantSize` if you do not need this information. (See "Data Not Wanted Constants" (page 112) for more information.)

extendedData
A pointer to a buffer to receive the extended data. Pass the constant `kPMDontWantData` if you do not want to read the data. (See "Data Not Wanted Constants" (page 112) for more information.)

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Your application typically needs to call the function `PMGetPageFormatExtendedData` two times in order to retrieve the extended page format data. The first time, pass the constant `kPMDontWantData` in the parameter *extendedData* to obtain the buffer size required for the extended data. Then allocate the buffer and call the function a second time to read the extended data into your buffer.

If you write a printing dialog extension for your application that stores data in the page format object, you use the function `PMGetPageFormatExtendedData` to retrieve the data associated with it.

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMSetPageFormatExtendedData` (page 100)

Declared In
`PMCore.h`

PMGetPageFormatPaper

Obtains the paper associated with a page format object.

```
OSStatus PMGetPageFormatPaper (
   PMPageFormat format,
   PMPaper *paper
);
```

Parameters

pageFormat

The page format object whose paper you want to obtain.

paper

A pointer to your PMPaper (page 109) variable. On return, the variable refers to a paper object that represents the paper associated with the specified page format. You should not release the paper object without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

PMCore.h

PMGetPageRange

Obtains the valid range of pages that can be printed.

```
OSStatus PMGetPageRange (
   PMPrintSettings printSettings,
   UInt32 *minPage,
   UInt32 *maxPage
);
```

Parameters

printSettings

The print settings object whose page range you want to obtain.

minPage

A pointer to your UInt32 variable. On return, the variable contains the minimum page number allowed.

maxPage

A pointer to your UInt32 variable. On return, the variable contains the maximum page number allowed.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The page range returned by the function PMGetPageRange is independent of the first and last page values returned by PMGetFirstPage (page 34) and PMGetLastPage (page 35). See PMSetPageRange for more information.

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMSetPageRange` (page 101)

Declared In
`PMCore.h`

PMGetPrintSettingsExtendedData

Obtains extended print settings data previously stored by your application.

```
OSStatus PMGetPrintSettingsExtendedData (
   PMPrintSettings printSettings,
   OSType dataID,
   UInt32 *size,
   void *extendedData
);
```

Parameters

printSettings

The print settings object whose extended data you want to obtain.

dataID

The unique 4-character code of the data to retrieve. This is typically your application's creator code. However, if your creator code is outside the ASCII 7-bit character range 0x20–0x7F, you need to use a different 4-character code.

size

A pointer to a value that specifies the size of the buffer you have allocated for the extended print settings data. On return, this variable contains the number of bytes read into the buffer or the size of the extended data. You can pass the constant `kPMDontWantSize` if you do not need this information. (See "Data Not Wanted Constants" (page 112) for more information.)

extendedData

A pointer to a buffer to receive the extended data. Pass the constant `kPMDontWantData` if you do not want to read the data. (See "Data Not Wanted Constants" (page 112) for more information.)

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Your application typically needs to call `PMGetPrintSettingsExtendedData` two times in order to retrieve the extended print settings data. The first time, pass the constant `kPMDontWantData` in the *extendedData* parameter to obtain the buffer size required for the extended data. Then allocate the buffer and call the function a second time to read the extended data into your buffer.

You may find it easier to use the functions `PMPrintSettingsSetValue` (page 74) and `PMPrintSettingsGetValue` (page 73) to store and retrieve user-defined data in a print settings object. If you use these functions, make sure that the custom keys you define for your private data do not conflict with other print settings keys.

Availability
Available in Mac OS X v10.0 and later.

See Also
PMSetPrintSettingsExtendedData (page 102)

Declared In
PMCore.h

PMGetScale

Obtains the scaling factor currently applied to the page and paper rectangles.

```
OSStatus PMGetScale (
   PMPageFormat pageFormat,
   double *scale
);
```

Parameters

pageFormat
The page format object whose scaling factor you want to obtain.

scale
A pointer to your double-precision variable. On return, the variable contains the scaling factor expressed as a percentage. For example, a value of 100.0 means 100 percent (that is, no scaling); a value of 50.0 means 50 percent scaling.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

See Also
PMSetScale (page 103)

Declared In
PMCore.h

PMGetUnadjustedPageRect

Obtains the imageable area or page rectangle, unaffected by orientation, resolution, or scaling.

```
OSStatus PMGetUnadjustedPageRect (
   PMPageFormat pageFormat,
   PMRect *pageRect
);
```

Parameters

pageFormat
The page format object whose unadjusted page rectangle you want to obtain.

pageRect
A pointer to your PMRect (page 111) data structure. On return, the structure contains the size of the page rectangle, in points. The page rectangle is the area of the page to which an application can draw. The coordinates for the upper-left corner of the page rectangle are (0,0). See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMCore.h`

PMGetUnadjustedPaperRect

Obtains the paper rectangle, unaffected by rotation, resolution, or scaling.

```
OSStatus PMGetUnadjustedPaperRect (
   PMPageFormat pageFormat,
   PMRect *paperRect
);
```

Parameters

pageFormat

The page format object whose unadjusted paper rectangle you want to obtain.

paperRect

A pointer to your `PMRect` (page 111) data structure. On return, the structure contains the physical size of the paper, in points. The coordinates of the upper-left corner of the paper rectangle are specified relative to the page rectangle. The coordinates of the upper-left corner of the page rectangle are always (0,0), which means the coordinates of the upper-left corner of the paper rectangle are always negative or (0,0). See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMCore.h`

PMPageFormatCreateDataRepresentation

Creates a data representation of a page format object.

```
OSStatus PMPageFormatCreateDataRepresentation (
    PMPageFormat pageFormat,
    CFDataRef *data,
    PMDataFormat format
);
```

Parameters

pageFormat

The page format object to convert.

data

A pointer to your CFDataRef variable. On return, the variable refers to a new Core Foundation data object that contains a representation of the specified page format object in the specified data format. You are responsible for releasing the data object.

format

A constant that specifies the format of the data representation. Supported values are:

- kPMDataFormatXMLDefault (compatible with all Mac OS X versions)
- kPMDataFormatXMLMinimal (approximately 3-5 times smaller; compatible with Mac OS X v10.5 and later)
- kPMDataFormatXMLCompressed (approximately 20 times smaller; compatible with Mac OS X v10.5 and later)

See "Data Representation Formats" (page 113) for a full description of these formats.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is typically used to convert a page format object into a data representation suitable for storage in a user document. For information about using a Core Foundation data object, see *CFData Reference*.

Before calling this function, you should call the function PMSessionValidatePageFormat (page 93) to make sure the page format object contains valid values.

Availability

Available in Mac OS X v10.5 and later.

See Also

PMPageFormatCreateWithDataRepresentation (page 42)

Declared In

PMCore.h

PMPageFormatCreateWithDataRepresentation

Creates a page format object from a data representation.

```
OSStatus PMPageFormatCreateWithDataRepresentation (
    CFDataRef data,
    PMPageFormat *pageFormat
);
```

Parameters

data

The data representation of a page format object. The data representation must have been previously created with the function PMPageFormatCreateDataRepresentation (page 41).

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a new page format object that contains the information stored in the specified data object. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is typically used to convert a data representation stored in a user document back into a page format object. For information about creating a Core Foundation data object from raw data, see *CFData Reference*.

After calling this function, you should call the function `PMSessionValidatePageFormat` (page 93) to make sure the page format object contains valid values.

Availability

Available in Mac OS X v10.5 and later.

See Also

`PMPageFormatCreateDataRepresentation` (page 41)

Declared In

`PMCore.h`

PMPageFormatGetPrinterID

Obtains the identifier of the formatting printer for a page format object.

```
OSStatus PMPageFormatGetPrinterID (
    PMPageFormat pageFormat,
    CFStringRef *printerID
);
```

Parameters

pageFormat

The page format object whose printer identifier you want to obtain.

printerID

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string that contains the identifier of the formatting printer for the specified page format object. If the page format object does not have that information, the variable is set to `NULL`. You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Page format objects can be created a number of different ways and some of them do not require a specific printer. If the printer ID is known, the printer is displayed in the Page Setup dialog's Format for pop-up menu. If the printer ID is not known, the default formatting printer is the generic Any Printer. The printing system provides default page and paper sizes for the generic printer.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPaperCreateCustom

Creates a custom paper object.

```
OSStatus PMPaperCreateCustom (
    PMPrinter printer,
    CFStringRef id,
    CFStringRef name,
    double width,
    double height,
    const PMPaperMargins *margins,
    PMPaper *paperP
);
```

Parameters

printer

A printer for which the specified paper size is appropriate.

id

A unique identifier for this custom paper. For example, you could create a UUID string and use it as the unique identifier.

name

The name to display to the user for this custom paper.

width

The width of the paper, in points.

height

The height of the paper, in points.

margins

A pointer to a `PMPaperMargins` (page 109) structure that specifies the unprintable margins of the paper, in points. The four values in the structure specify the top, left, bottom, and right imageable area margins of the paper.

paperP

A pointer to your `PMPaper` (page 109) variable. On return, the variable refers to a new custom paper object. You are responsible for releasing the paper object with the function `PMRelease` (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function creates a custom paper object appropriate for the specified printer. Custom papers are treated differently than built-in papers by the printing system. To obtain one of the available built-in papers for a given printer, you can use the function `PMPrinterGetPaperList` (page 61).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPaperCreateLocalizedName

Obtains the localized name for a given paper.

```
OSStatus PMPaperCreateLocalizedName (
    PMPaper paper,
    PMPrinter printer,
    CFStringRef *paperName
);
```

Parameters

paper

The paper whose localized name you want to obtain.

printer

The printer for which the localization should be performed.

paperName

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string that contains the localized name of the paper. This name is appropriate to display in the user interface. If an error occurs, the variable is set to `NULL`. You are responsible for releasing the string.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Not all printers have the same way of referring to a given paper. Generally, if you want to obtain the name of a paper, you want to localize the paper name for a particular printer. For example, if you were displaying a list of papers for a given printer, you would want the paper names to be localized for that printer.

Special Considerations

In Mac OS X v10.5 and later, Apple recommends using this function instead of `PMPaperGetName` (page 47).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPaperGetHeight

Obtains the height of the sheet of paper represented by a paper object.

```
OSStatus PMPaperGetHeight (
   PMPaper paper,
   double *paperHeight
);
```

Parameters

paper

The paper whose height you want to obtain.

paperHeight

A pointer to your double-precision variable. On return, the variable contains the height of the specified paper, in points.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPaperGetID

Obtains the identifier of a paper object.

```
OSStatus PMPaperGetID (
   PMPaper paper,
   CFStringRef *paperID
);
```

Parameters

paper

The paper whose identifier you want to obtain.

paperID

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string containing the unique identifier for this paper. You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPaperGetMargins

Obtains the margins describing the unprintable area of the sheet represented by a paper object.

```
OSStatus PMPaperGetMargins (
   PMPaper paper,
   PMPaperMargins *paperMargins
);
```

Parameters

paper

The paper whose margins you want to obtain.

paperMargins

A pointer to your `PMPaperMargins` (page 109) structure. On return, the structure contains the unprintable margins of the specified paper, in points. The four values in the structure specify the top, left, bottom, and right imageable area margins of the paper.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPaperGetName

Obtains the name for a given paper.

```
OSStatus PMPaperGetName (
   PMPaper paper,
   CFStringRef *paperName
);
```

Parameters

paper

The paper whose name you want to obtain.

paperName

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string containing the name for this paper. This name identifies the paper in the user interface. You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Special Considerations

This function does not necessarily return a paper name that's localized for a given printer. In Mac OS X v10.5 and later, instead of using this function, Apple recommends using the function `PMPaperCreateLocalizedName` (page 44).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPaperGetPPDPaperName

Obtains the PPD paper name for a given paper.

```
OSStatus PMPaperGetPPDPaperName (
    PMPaper paper,
    CFStringRef *paperName
);
```

Parameters

paper

The paper whose PPD paper name you want to obtain.

paperName

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string that contains the PPD paper name for the specified paper. If an error occurs, the variable is set to `NULL`. You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The Mac OS X printing system uses a PostScript Printer Description (PPD) file to describe a given printer and print queue for that printer. The PPD paper name is the name that uniquely identifies a given paper for the printer to which the paper corresponds. To obtain a list of papers for a given printer, use the function `PMPrinterGetPaperList` (page 61).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPaperGetPrinterID

Obtains the printer ID of the printer to which a given paper corresponds.

```
OSStatus PMPaperGetPrinterID (
    PMPaper paper,
    CFStringRef *printerID
);
```

Parameters

`paper`

The paper whose printer ID you want to obtain.

`printerID`

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string that contains the printer ID for the specified paper. If an error occurs, the variable is set to `NULL`. You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Not all papers have a printer ID associated with them. If the printer ID is known, the printer is displayed in the Page Setup dialog's Format for pop-up menu. If the printer ID is not known, the default formatting printer is the generic Any Printer. The printing system provides default paper sizes for the generic printer.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPaperGetWidth

Obtains the width of the sheet of paper represented by a paper object.

```
OSStatus PMPaperGetWidth (
   PMPaper paper,
   double *paperWidth
);
```

Parameters

paper

The paper whose width you want to obtain.

paperWidth

A pointer to your double-precision variable. On return, the variable contains the width of the specified paper, in points.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPaperIsCustom

Returns a Boolean value indicating whether a specified paper is a custom paper.

```
Boolean PMPaperIsCustom (
    PMPaper paper
);
```

Parameters

paper

The paper you're querying to determine whether it's a custom paper.

Return Value

If `true`, the specified paper is a custom paper; otherwise, `false`.

Discussion

You can create a custom paper with the function `PMPaperCreateCustom` (page 43).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPresetCopyName

Obtains the localized name for a preset.

```
OSStatus PMPresetCopyName (
   PMPreset preset,
   CFStringRef *name
);
```

Parameters

preset

The preset object whose localized name you want to obtain. You can use the function PMPrinterCopyPresets (page 53) to obtain the presets for a given printer.

paperID

A pointer to your CFStringRef variable. On return, the variable refers to a Core Foundation string containing the localized name of the specified preset. You are responsible for releasing the string.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

PMCore.h

PMPresetCreatePrintSettings

Creates a print settings object with settings that correspond to a preset.

```
OSStatus PMPresetCreatePrintSettings (
   PMPreset preset,
   PMPrintSession session,
   PMPrintSettings *printSettings
);
```

Parameters

preset

The preset whose settings you want to obtain. You can use the function PMPrinterCopyPresets (page 53) to obtain the presets for a given printer.

session

The session you use to present the Print dialog.

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a print settings object with settings that correspond to the specified preset. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.3 and later.

Declared In

PMCore.h

PMPresetGetAttributes

Obtains the attributes of a preset.

```
OSStatus PMPresetGetAttributes (
   PMPreset preset,
   CFDictionaryRef *attributes
);
```

Parameters

preset

The preset whose attributes you want to obtain. You can use the function `PMPrinterCopyPresets` (page 53) to obtain the presets for a given printer.

attributes

A pointer to your `CFDictionaryRef` variable. On return, the variable refers to a Core Foundation dictionary containing the attributes of the specified preset, or `NULL` if the attributes could not be obtained. For more information about these attributes, see the Discussion. You should not release this dictionary without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

A preset has associated with it a dictionary containing the preset identifier, the localized name, and a description of the environment for which the preset is intended. In addition to these standard attributes, the preset you specify may contain additional attributes that reflect custom print settings.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPrinterCopyDescriptionURL

Obtains the URL of the description file for a given printer.

```
OSStatus PMPrinterCopyDescriptionURL (
   PMPrinter printer,
   CFStringRef descriptionType,
   CFURLRef *fileURL
);
```

Parameters

printer

The printer whose description file you want to obtain.

descriptionType

A constant that specifies the desired printer description file type. Currently, you must pass the constant `kPMPPDDescriptionType`.

fileURL

A pointer to your `CFURLRef` variable. On return, the variable refers to a Core Foundation URL that specifies the location of the file that contains a description of the specified printer. You are responsible for releasing the URL. If an error occurs, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can use this function to locate the PostScript printer description (PPD) file for a printer.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`PMCore.h`

PMPrinterCopyDeviceURI

Obtains the device URI of a given printer.

```
OSStatus PMPrinterCopyDeviceURI (
   PMPrinter printer,
   CFURLRef *deviceURI
);
```

Parameters

printer

The printer whose device URI you want to obtain.

deviceURI

A pointer to your `CFURLRef` variable. On return, the variable refers to a Core Foundation URL that specifies the printer's device URI. You are responsible for releasing the URL. If an error occurs, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The device URI of a printer describes how to communicate with the device. For some devices, it also includes a unique identifier for the device.

Availability

Available in Mac OS X v10.4 and later.

Related Sample Code

PMPrinterTest

Declared In

`PMCore.h`

PMPrinterCopyHostName

Obtains the name of the server hosting the print queue for a given printer.

```
OSStatus PMPrinterCopyHostName (
    PMPrinter printer,
    CFStringRef *hostNameP
);
```

Parameters

printer

The printer whose print queue host name you want to obtain.

hostNameP

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string containing the name of the specified printer's server. You are responsible for releasing the string.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is typically used to obtain the name of the computer that hosts a shared printer, possibly for display in a user interface. In Mac OS X v10.5 and later, the typical way that users browse and communicate with a shared printer creates a local print queue and `PMPrinterCopyHostName` for such a print queue will return the name of the local host.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPrinterCopyPresets

Obtains a list of print settings presets for a printer.

```
OSStatus PMPrinterCopyPresets (
   PMPrinter printer,
   CFArrayRef *presetList
);
```

Parameters

printer

The printer whose presets you want to obtain.

presetList

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array containing the presets for the specified printer. Each element in the array is an object of type `PMPreset` (page 109). You are responsible for releasing the array.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

A printer may have associated with it a list of preset settings. Each setting is optimized for a particular printing situation. This function returns all of the presets for a given printer. To obtain more information about a particular preset, you can use the function `PMPresetGetAttributes` (page 51). To create a print settings object that contains the settings of a preset, call `PMPresetCreatePrintSettings` (page 50).

Availability

Available in Mac OS X v10.3 and later.

Declared In

PMCore.h

PMPrinterCreateFromPrinterID

Creates a printer object from a print queue identifier.

```
PMPrinter PMPrinterCreateFromPrinterID (
   CFStringRef printerID
);
```

Parameters

printerID

The unique identifier of a print queue.

Return Value

A new printer object, or NULL if no print queue is available with the specified identifier. You are responsible for releasing the printer object with the function PMRelease (page 76).

Discussion

This function is typically used to re-create a printer object using the print queue ID obtained by a call to PMPrinterGetID at an earlier time. If the print queue is deleted after obtaining the ID, this function returns NULL for that ID.

Availability

Available in Mac OS X v10.4 and later.

See Also

PMPrinterGetID (page 56)

Declared In

PMCore.h

PMPrinterGetCommInfo

Obtains information about the communication channel for a printer.

```
OSStatus PMPrinterGetCommInfo (
   PMPrinter printer,
   Boolean *supportsTransparentP,
   Boolean *supportsEightBitP
);
```

Parameters

printer

The printer whose information you want to obtain.

supportsTransparentP

A pointer to your Boolean variable. On return, true indicates that the communication channel to the specified printer supports bytes in the range 0x0–0x1F; otherwise, false.

supportsEightBitP

A pointer to your Boolean variable. On return, true indicates that the communication channel to the specified printer supports bytes in the range 0x80–0xFF; otherwise, false.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
This function is typically relevant only to PostScript printers. All PostScript printers, regardless of what communications channel is used to send data to them, support data in the range 0x20–0x7F. Many communications channels can support data outside this range. You can use this function to determine whether the communications channel to the specified printer also supports bytes in the ranges 0x0–0x1F and 0x80–0xFF.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMCore.h`

PMPrinterGetDriverCreator

Obtains the creator of the driver associated with the specified printer.

```
OSStatus PMPrinterGetDriverCreator (
   PMPrinter printer,
   OSType *creator
);
```

Parameters

`printer`
The printer whose driver creator you want to obtain.

`creator`
On return, the 4-byte creator code of the driver (for example, `'APPL'` for an Apple printer driver).

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
This function is not recommended because it makes your application driver-dependent.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMCore.h`

PMPrinterGetDriverReleaseInfo

Obtains version information for the driver associated with the specified printer.

```
OSStatus PMPrinterGetDriverReleaseInfo (
   PMPrinter printer,
   VersRec *release
);
```

Parameters

printer

The printer whose driver version you want to obtain.

release

A pointer to your VersRec data structure. On return, the structure contains the driver's short and long version strings and country code.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is not recommended because it makes your application driver-dependent. If you do use this function, you must call it between the creation and release of a printing session. See the function PMCreateSession (page 31).

Availability

Available in Mac OS X v10.0 and later.

Declared In

PMCore.h

PMPrinterGetID

Returns the unique identifier of a printer.

```
CFStringRef PMPrinterGetID (
   PMPrinter printer
);
```

Parameters

printer

The printer whose identifier you want to obtain.

Return Value

The identifier of the specified printer. You should not release the string without first retaining it. If the specified printer is not valid, this function returns NULL.

Discussion

You can use the function PMPrinterGetID to capture information about a printer for later use. To create a printer object from a printer ID returned by this function, use the function PMPrinterCreateFromPrinterID (page 54).

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

PMPrinterTest

Declared In

PMCore.h

PMPrinterGetIndexedPrinterResolution

Obtains a resolution setting based on an index into the range of settings supported by the specified printer.

```
OSStatus PMPrinterGetIndexedPrinterResolution (
   PMPrinter printer,
   UInt32 index,
   PMResolution *resolutionP
);
```

Parameters

printer

The printer whose resolution you want to obtain.

index

An index into the range of resolution settings supported by the specified printer. Index values begin at 1.

res

A pointer to your `PMResolution` (page 111) data structure. On return, the structure contains the printer resolution setting associated with the index value.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. Before you call this function, you must call the function `PMPrinterGetPrinterResolutionCount` (page 61) to obtain the number of resolution settings supported by the specified printer.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMCore.h`

PMPrinterGetLanguageInfo

Obtains information about the imaging language for the specified printer.

```
OSStatus PMPrinterGetLanguageInfo (
   PMPrinter printer,
   PMLanguageInfo *info
);
```

Parameters

printer

The printer whose imaging language information you want to obtain.

info

A pointer to your `PMLanguageInfo` (page 107) data structure. On return, the structure contains the printer's language level, version, and release information. The format of the returned data uses the syntax of the PostScript language.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion
The function `PMPrinterGetLanguageInfo` is useful only for PostScript printers. You must call this function between the creation and release of a printing session.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMCore.h`

PMPrinterGetLocation

Returns the location of a printer.

```
CFStringRef PMPrinterGetLocation (
   PMPrinter printer
);
```

Parameters

printer

The printer whose location you want to obtain.

Return Value
The location of the specified printer. You should not release the string without first retaining it. If the printer is not valid, this function returns `NULL`.

Discussion
The location of a printer is specified when a user creates a print queue for the printer. In some cases, the printing system automatically determines the location. For example, the location may be set to “Local Zone”. The user creating the print queue can also set the location.

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
PMPrinterTest

Declared In
`PMCore.h`

PMPrinterGetMakeAndModelName

Obtains the manufacturer and model name of the specified printer.

```
OSStatus PMPrinterGetMakeAndModelName (
   PMPrinter printer,
   CFStringRef *makeAndModel
);
```

Parameters

printer

The printer whose manufacturer and model name you want to obtain.

makeAndModel

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string containing the manufacturer and model name of the specified printer. You should not release the string without first retaining it. If an error occurs, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

PMPrinterTest

Declared In

`PMCore.h`

PMPrinterGetMimeTypes

Obtains a list of MIME content types supported by a printer using the specified print settings.

```
OSStatus PMPrinterGetMimeTypes (
   PMPrinter printer,
   PMPrintSettings settings,
   CFArrayRef *mimeTypes
);
```

Parameters

printer

The printer whose supported MIME types you want to obtain.

settings

The print settings for the print job. The print settings object contains the job destination, which affects the available types. This parameter may be `NULL`.

mimeTypes

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array containing the MIME types supported by the specified printer. Each element in the array is a Core Foundation string. You should not release the array without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function retrieves the types of data that can be submitted to a printer with the specified print settings; for example, `application/pdf`. This function is typically used in conjunction with the function `PMPrinterPrintWithFile` (page 65).

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPrinterGetName

Returns the human-readable name of a printer.

```
CFStringRef PMPrinterGetName (
   PMPrinter printer
);
```

Parameters

printer

The printer whose name you want to obtain.

Return Value

The name of the specified printer. This name identifies the printer in the user interface. You should not release the string without first retaining it.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

PMPrinterTest

Declared In

`PMCore.h`

PMPrinterGetOutputResolution

Obtains the printer hardware output resolution for the specified print settings.

```
OSStatus PMPrinterGetOutputResolution (
    PMPrinter printer,
    PMPrintSettings printSettings,
    PMResolution *resolutionP
);
```

Parameters

printer

The printer whose output resolution you want to obtain.

printSettings

The print settings you want to use.

resolutionP

A pointer to your `PMResolution` (page 111) structure. On return, the structure contains the output resolution of the specified printer in pixels per inch.

Return Value

A result code. If the resolution cannot be reliably determined, this function returns an error.

Discussion

Some printers allow programmatic control of their hardware output resolution on a print job basis. The hardware resolution is determined by the combination of printer and print settings used for the print job. This function returns the best guess as to what printer resolution setting will be used for the destination print job.

Most applications do not need to use this function because they draw the same content regardless of the destination device. For those few applications that do adjust their drawing based on the output device, they should only do so when the print job destination is `kPMDestinationPrinter` or `kPMDestinationFax`. You can use the function `PMSessionGetDestinationType` to determine the destination for a print job.

This function should be used after displaying the Print dialog to the user so that it correctly reflects changes in print settings performed prior to printing.

Availability

Available in Mac OS X v10.5 and later.

See Also

`PMPrinterSetOutputResolution` (page 68)

Declared In

`PMCore.h`

PMPrinterGetPaperList

Obtains the list of papers available for a printer.

```
OSStatus PMPrinterGetPaperList (
   PMPrinter printer,
   CFArrayRef *paperList
);
```

Parameters

`printer`

The printer whose paper list you want to obtain.

`paperList`

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array containing the paper list for the specified printer. Each element in the array is an object of type `PMPaper` (page 109). You should not release the array without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function obtains a list of the papers that a given printer claims to support. The paper list does not include any custom paper sizes that may be available.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPrinterGetPrinterResolutionCount

Obtains the number of resolution settings supported by the specified printer.

```
OSStatus PMPrinterGetPrinterResolutionCount (
   PMPrinter printer,
   UInt32 *countP
);
```

Parameters

printer

The printer whose number of resolution settings you want to obtain.

count

A pointer to your `UInt32` variable. On return, the variable contains the number of resolutions that are supported for the specified printer.

Return Value

A result code. The result code `kPMNotImplemented` indicates that the printer driver does not support multiple resolution settings.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMCore.h`

PMPrinterGetState

Obtains the current state of the print queue for a printer.

```
OSStatus PMPrinterGetState (
   PMPrinter printer,
   PMPrinterState *state
);
```

Parameters

printer

The printer whose queue state you want to obtain.

state

A pointer to your `PMPrinterState` variable. On return, the variable contains a constant that indicates the current state of the print queue for the specified printer. Supported values are:

- `kPMPrinterIdle` (queue is idle)
- `kPMPrinterProcessing` (queue is processing a job)
- `kPMPrinterStopped` (queue is stopped)

See "Print Queue States" (page 127) for a complete description of these constants.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

PMPrinterTest

Declared In
`PMCore.h`

PMPrinterIsDefault

Returns a Boolean value indicating whether a printer is the default printer for the current user.

```
Boolean PMPrinterIsDefault (
   PMPrinter printer
);
```

Parameters

`printer`

The printer you're querying to determine whether it is the default printer.

Return Value
If `true`, the specified printer is the default printer for the current user; otherwise, `false`.

Discussion
The default printer is the printer selected by default in the Print dialog.

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
PMPrinterTest

Declared In
`PMCore.h`

PMPrinterIsFavorite

Returns a Boolean value indicating whether a printer is in the user's list of favorite printers.

```
Boolean PMPrinterIsFavorite (
   PMPrinter printer
);
```

Parameters

`printer`

The printer you're looking for in the favorite printer list.

Return Value
If `true`, the specified printer is in the user's list of favorite printers; otherwise, `false`.

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
PMPrinterTest

Declared In
`PMCore.h`

PMPrinterIsPostScriptCapable

Returns a Boolean value indicating whether a printer is PostScript capable.

```
Boolean PMPrinterIsPostScriptCapable (
   PMPrinter printer
);
```

Parameters

printer

The printer you're querying to determine whether it's PostScript capable.

Return Value

If `true`, the specified printer is a PostScript capable printer; otherwise, `false`.

Discussion

A printer that is PostScript capable is not necessarily a PostScript printer. The Mac OS X printing system can render PostScript content on non-PostScript printers.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`PMCore.h`

PMPrinterIsPostScriptPrinter

Determines whether a printer is a PostScript printer.

```
OSStatus PMPrinterIsPostScriptPrinter (
    PMPrinter printer,
    Boolean *isPSPrinter
);
```

Parameters

printer

The printer you're querying to determine whether it's a PostScript printer.

isPSPrinter

A pointer to your Boolean variable. On return, `true` indicates that the specified printer is a PostScript printer; otherwise, `false`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

A printer is a PostScript printer if the printer driver takes PostScript directly.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPrinterIsRemote

Indicates whether a printer is hosted by a remote print server.

```
OSStatus PMPrinterIsRemote (
    PMPrinter printer,
    Boolean *isRemoteP
);
```

Parameters

printer

The printer you're querying to determine whether it is hosted by a remote print server.

isRemoteP

A pointer to your Boolean variable. On return, `true` indicates that the printer is hosted by a remote print server; otherwise, `false`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

If this function returns `true`, the printer is hosted by a remote print server and the printer can be considered a shared printer.

In Mac OS X, the typical way that users create a print queue for a shared printer is by browsing. Print queues for shared printers that are created by browsing are marked as remote queues, and `PMPrinterIsRemote` returns `true` for such printers. However, expert users can create a local queue for a remote printer manually, and such a printer does not appear to be remote printer.

Whether a printer is remote is derived from the CUPS printer-type attribute for the print queue.

Availability

Available in Mac OS X v10.3 and later.

Related Sample Code

PMPrinterTest

Declared In

`PMCore.h`

PMPrinterPrintWithFile

Submits a print job to a specified printer using a file that contains print data.

```
OSStatus PMPrinterPrintWithFile (
   PMPrinter printer,
   PMPrintSettings settings,
   PMPageFormat format,
   CFStringRef mimeType,
   CFURLRef fileURL
);
```

Parameters

printer

The destination printer.

settings

The print settings for the print job.

format

The physical page size and orientation with which the document should be printed. This parameter can be `NULL`.

mimeType

The MIME type of the data to be printed. If this parameter is `NULL`, the MIME type will be determined automatically. You can obtain a list of the MIME types supported by a given printer using the function `PMPrinterGetMimeTypes` (page 59).

fileURL

The URL of the file that supplies the print data.

Return Value

A result code. See "Core Printing Result Codes" (page 131). If the specified printer cannot handle the file's MIME type, a non-zero error code is returned.

Discussion

This function can fail if the specified printer cannot handle the file's MIME type. Use the function `PMPrinterGetMimeTypes` (page 59) to check whether a MIME type is supported.

Availability

Available in Mac OS X v10.3 and later.

See Also

`PMPrinterPrintWithProvider` (page 66)

Declared In

`PMCore.h`

PMPrinterPrintWithProvider

Submits a print job to a specified printer using a Quartz data provider to obtain the print data.

```
OSStatus PMPrinterPrintWithProvider (
   PMPrinter printer,
   PMPrintSettings settings,
   PMPageFormat format,
   CFStringRef mimeType,
   CGDataProviderRef provider
);
```

Parameters

printer

The destination printer.

settings

The print settings for the print job.

format

The physical page size and orientation with which the document should be printed. This parameter can be `NULL`.

mimeType

The MIME type of the data to be printed. This parameter cannot be `NULL`. If you want automatic typing, use the function `PMPrinterPrintWithFile` (page 65) instead. You can obtain a list of the MIME types supported by a given printer using the function `PMPrinterGetMimeTypes` (page 59).

provider

The data provider that supplies the print data.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function can fail if the specified printer cannot handle the data provider's MIME type. Use the function `PMPrinterGetMimeTypes` (page 59) to check whether a MIME type is supported.

Special Considerations

In Mac OS X v10.4 and earlier, this function is not implemented and returns the error code –1 when called. You can write your print data to a file and use `PMPrinterPrintWithFile` instead.

Availability

Available in Mac OS X v10.3 and later.

See Also

`PMPrinterPrintWithFile` (page 65)

Declared In

`PMCore.h`

PMPrinterSetDefault

Sets the default printer for the current user.

```
OSStatus PMPrinterSetDefault (
    PMPrinter printer
);
```

Parameters

printer

The printer to set as the default printer.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The default printer is the printer selected by default in the Print dialog.

This function is rarely used. Most applications do not set the default printer directly, but instead let the user choose the default printer in the Print & Fax preference pane of System Preferences.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPrinterSetOutputResolution

Sets the print settings to reflect the specified printer hardware output resolution.

```
OSStatus PMPrinterSetOutputResolution (
    PMPrinter printer,
    PMPrintSettings printSettings,
    const PMResolution *resolutionP
);
```

Parameters

printer

The printer whose output resolution you want to change.

printSettings

The print settings object used for the print job.

resolutionP

A pointer to a `PMResolution` (page 111) structure that specifies the desired resolution in pixels per inch.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Some printers allow programmatic control of their hardware output resolution on a print job basis. The hardware resolution is determined by the combination of printer and print settings used for the print job. This function configures the print settings to the closest resolution setting that can be used for the destination print job. Note that not all printers allow control of their resolution setting.

This function is rarely used. Most applications do not set the output resolution but instead use the setting supplied by the user in the Print dialog.

Availability

Available in Mac OS X v10.5 and later.

See Also

`PMPrinterGetOutputResolution` (page 60)

Declared In

`PMCore.h`

PMPrinterWritePostScriptToURL

Converts an input file of the specified MIME type to printer-ready PostScript for a destination printer.

```
OSStatus PMPrinterWritePostScriptToURL (
    PMPrinter printer,
    PMPrintSettings settings,
    PMPageFormat format,
    CFStringRef mimeType,
    CFURLRef sourceFileURL,
    CFURLRef destinationFileURL
);
```

Parameters

printer

The destination printer for which printer-ready PostScript will be generated.

settings

The print settings for the print job.

format

The page format specifying the physical page size and orientation on which the document should be printed.

mimeType

The MIME type of the file to be printed. If you pass `NULL`, the file is typed automatically. You can obtain a list of the MIME types supported by a given printer using the function `PMPrinterGetMimeTypes` (page 59).

sourceFileURL

A URL specifying the input file to be converted to printer-ready PostScript data. Only file-based URLs are supported.

destinationFileURL

A URL specifying the destination file to be created. If the file already exists, it will be overwritten. Only file-based URLs are supported.

Return Value

A result code. If the printing system cannot convert the input MIME type to PostScript, this function fails and returns an error.

Discussion

This function is synchronous; the conversion of the input file to PostScript is performed before the function returns. This can take a significant amount of time for longer documents. You may want to perform this operation on a thread other than the main application thread or fork a separate process for this purpose.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPrintSettingsCopyAsDictionary

Creates a dictionary that contains the settings in a print settings object.

```
OSStatus PMPrintSettingsCopyAsDictionary (
    PMPrintSettings printSettings,
    CFDictionaryRef *settingsDictionary
);
```

Parameters

printSettings

The print settings object with the desired settings.

settingsDictionary

A pointer to your CFDictionaryRef variable. On return, the variable refers to a Core Foundation dictionary that contains the settings in the specified print settings object. Some of the keys in this dictionary are currently defined in PMTicket.h; other keys are user-defined. You are responsible for releasing the dictionary. If an error occurs, the variable is set to NULL.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Most developers have no need to use this function. However, one way this function might be useful would be to enumerate all the entries in a print settings object for inspection.

Availability

Available in Mac OS X v10.5 and later.

Declared In

PMCore.h

PMPrintSettingsCopyKeys

Obtains the keys for items in a print settings object.

```
OSStatus PMPrintSettingsCopyKeys (
    PMPrintSettings printSettings,
    CFArrayRef *settingsKeys
);
```

Parameters

printSettings

The print settings object with the desired keys.

settingsKeys

A pointer to your CFArrayRef variable. On return, the variable refers to a Core Foundation array that contains the keys for items in the specified print settings object. Each of these keys may be passed to the function PMPrintSettingsGetValue to obtain a value. You are responsible for releasing the array. If an error occurs, the variable is set to NULL.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function provides an array of the keys in a print settings object. You could get the values for the keys in the array with PMPrintSettingsGetValue (page 73), or use the keys to look up the values in the dictionary returned by PMPrintSettingsCopyAsDictionary (page 69).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMPrintSettingsCreateDataRepresentation

Creates a data representation of a print settings object.

```
OSStatus PMPrintSettingsCreateDataRepresentation (
    PMPrintSettings printSettings,
    CFDataRef *data,
    PMDataFormat format
);
```

Parameters

printSettings

The print settings object to convert.

data

A pointer to your `CFDataRef` variable. On return, the variable refers to a new Core Foundation data object that contains a representation of the specified print settings object in the specified data format. You are responsible for releasing the data object.

format

A constant that specifies the format of the data representation. Supported values are:

- `kPMDataFormatXMLDefault` (compatible with all Mac OS X versions)
- `kPMDataFormatXMLMinimal` (approximately 3-5 times smaller; compatible with Mac OS X v10.5 and later)
- `kPMDataFormatXMLCompressed` (approximately 20 times smaller; compatible with Mac OS X v10.5 and later)

See "Data Representation Formats" (page 113) for a full description of these formats.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is typically used to convert a print settings object into a data representation suitable for storage in a user document. For information about using a Core Foundation data object, see *CFData Reference*.

Before calling this function, you should call the function `PMSessionValidatePrintSettings` (page 94) to make sure the print settings object contains valid values.

Apple recommends that you do not reuse the print settings information if the user prints the document again. The information supplied by the user in the Print dialog should pertain to the document only while the document prints, so there is no need to save the print settings object.

Availability

Available in Mac OS X v10.5 and later.

See Also

`PMPrintSettingsCreateWithDataRepresentation` (page 72)

Declared In

`PMCore.h`

PMPrintSettingsCreateWithDataRepresentation

Creates a print settings object from a data representation.

```
OSStatus PMPrintSettingsCreateWithDataRepresentation (
    CFDataRef data,
    PMPrintSettings *printSettings
);
```

Parameters

data

The data representation of a print settings object. The data representation must have been previously created with the function PMPrintSettingsCreateDataRepresentation (page 71).

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a new print settings object that contains the printing information stored in the specified data object. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is typically used to convert a data representation stored in a user document back into a print settings object. For information about creating a Core Foundation data object from raw data, see *CFData Reference*.

After calling this function, you should call the function PMSessionValidatePrintSettings (page 94) to make sure the print settings object contains valid values.

Availability

Available in Mac OS X v10.5 and later.

See Also

PMPrintSettingsCreateDataRepresentation (page 71)

Declared In

PMCore.h

PMPrintSettingsGetJobName

Obtains the name of a print job.

```
OSStatus PMPrintSettingsGetJobName (
   PMPrintSettings printSettings,
   CFStringRef *name
);
```

Parameters

printSettings

The print settings for the current print job.

name

A pointer to your `CFStringRef` variable. On return, the variable refers to a Core Foundation string containing the name of the print job. This is the same job name you set using the function `PMPrintSettingsSetJobName` (page 74). You should not release the string without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

See Also

`PMPrintSettingsSetJobName` (page 74)

Declared In

`PMCore.h`

PMPrintSettingsGetValue

Obtains the value of a setting in a print settings object.

```
OSStatus PMPrintSettingsGetValue (
    PMPrintSettings printSettings,
    CFStringRef key,
    CFTypeRef *value
);
```

Parameters

printSettings

The print settings object you want to access.

key

A string constant that specifies the key for the desired setting. Some keys are currently defined in `PMTicket.h`; other keys are user-defined.

value

A pointer to your Core Foundation variable. On return, the variable refers to a Core Foundation object that corresponds to the specified key. If no corresponding object exists, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function, together with the function `PMPrintSettingsSetValue`, makes it possible to access print settings directly.

Availability

Available in Mac OS X v10.4 and later.

See Also

`PMPrintSettingsSetValue` (page 74)

Declared In

`PMCore.h`

PMPrintSettingsSetJobName

Specifies the name of a print job.

```
OSStatus PMPrintSettingsSetJobName (
   PMPrintSettings printSettings,
   CFStringRef name
);
```

Parameters

printSettings

The print settings object whose job name you want to set.

name

The new name for the print job.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

If you're using the Print dialog, you should call this function before presenting the dialog. You are strongly encouraged to create a print job name that's meaningful to the user and use this function to set the name; this produces the best user experience. If you do not specify the print job name, the printing system creates an appropriate job name for you.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.4 and later.

See Also

`PMPrintSettingsGetJobName` (page 72)

Declared In

`PMCore.h`

PMPrintSettingsSetValue

Stores the value of a setting in a print settings object.

```
OSStatus PMPrintSettingsSetValue (
    PMPrintSettings printSettings,
    CFStringRef key,
    CFTypeRef value,
    Boolean locked
);
```

Parameters

printSettings

The print settings object you want to update.

key

A string constant that specifies the key for the desired setting. Some keys are currently defined in `PMTicket.h`; other keys are user-defined.

value

A Core Foundation object that corresponds to the specified key. If you pass NULL, any existing setting for the specified key is removed.

locked

If true, the item being set should be locked; otherwise, false. Currently, you should always pass false.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function makes it possible to add, change, or remove print settings directly. Print settings are stored as key-value pairs. The keys are Core Foundation strings and the corresponding values are Core Foundation objects.

You can use this function to store user-defined data in a print settings object. You should make sure that the custom keys you define for your private data do not conflict with any other keys in the object. Each data item you store needs to be a Core Foundation object. You can use the function PMPrintSettingsGetValue to retrieve your private data.

If you call this function after initiating a print job (for example, by calling PMSessionBeginCGDocument), the change is ignored for the current job.

Availability

Available in Mac OS X v10.4 and later.

See Also

PMPrintSettingsGetValue (page 73)

Declared In

PMCore.h

PMPrintSettingsToOptions

Converts print settings into a CUPS options string.

```
OSStatus PMPrintSettingsToOptions (
    PMPrintSettings settings,
    char **options
);
```

Parameters

settings

The print settings to convert.

options

A pointer to a C string. On return, a CUPS options string describing the print settings, or NULL if the print settings could not be converted. The function allocates storage for the string. You are responsible for freeing the storage.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function creates a CUPS options string that captures the data in the specified print settings object. In Mac OS X v10.5 and later, Apple recommends that you use the `PMPrintSettingsToOptionsWithPrinterAndPageFormat` (page 76) function instead.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

PMPrintSettingsToOptionsWithPrinterAndPageFormat

Converts print settings and page format data into a CUPS options string for a specified printer.

```
OSStatus PMPrintSettingsToOptionsWithPrinterAndPageFormat (
    PMPrintSettings settings,
    PMPrinter printer,
    PMPageFormat pageFormat,
    char **options
);
```

Parameters

`settings`

The print settings to convert.

`printer`

The printer to use for converting the print settings. This parameter must not be `NULL`.

`pageFormat`

The page format to convert, or `NULL` to specify default page format data.

`options`

A pointer to a C string. On return, a CUPS option string with the specified print settings and page format data, or `NULL` if the data could not be converted. The function allocates storage for the string. You are responsible for freeing the storage.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function creates a CUPS options string for the destination printer that captures the data in the specified print settings and page format objects. For example, you could pass this string to the function `PMWorkflowSubmitPDFWithOptions` (page 104) to submit a PDF file for workflow processing. You could also use the options string to run a CUPS filter directly.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`PMCore.h`

PMRelease

Releases a printing object by decrementing its reference count.

```
OSStatus PMRelease (
   PMObject object
);
```

Parameters

object

The printing object you want to release.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Your application should use the PMRelease function to release any printing objects it creates or retains. When an object's reference count reaches 0, the object is deallocated.

For example, to terminate a printing session created with the function PMCreateSession (page 31), pass the associated PMPrintSession (page 110) object to PMRelease. To release printing objects created with the functions PMCreatePageFormat (page 29) and PMCreatePrintSettings (page 30), pass the associated PMPageFormat and PMPrintSettings objects to PMRelease.

Availability

Available in Mac OS X v10.0 and later.

See Also

PMRetain (page 77)

Related Sample Code

CarbonSketch

Declared In

PMCore.h

PMRetain

Retains a printing object by incrementing its reference count.

```
OSStatus PMRetain (
   PMObject object
);
```

Parameters

object

The printing object you want to retain.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You should retain a printing object when you receive it from elsewhere (that is, you did not create or copy it) and you want it to persist. If you retain a printing object, you are responsible for releasing it. (See PMRelease.) You can use the function PMRetain to increment a printing object's reference count so that multiple threads or routines can use the object without the risk of another thread or routine deallocating the object.

Availability

Available in Mac OS X v10.0 and later.

See Also
`PMRelease` (page 76)

Declared In
`PMCore.h`

PMServerCreatePrinterList

Creates a list of printers available to a print server.

```
OSStatus PMServerCreatePrinterList (
   PMServer server,
   CFArrayRef *printerList
);
```

Parameters

server
: The print server whose printers you want to obtain. To specify the local print server, pass the constant `kPMServerLocal`. Currently, you may specify only the local print server.

printerList
: A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array containing the printers available to the specified print server. Each element in the array is a `PMPrinter` object. You are responsible for releasing the array.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
PMPrinterTest

Declared In
`PMCore.h`

PMServerLaunchPrinterBrowser

Launches the printer browser to browse the printers available for a print server.

```
OSStatus PMServerLaunchPrinterBrowser (
    PMServer server,
    CFDictionaryRef options
);
```

Parameters

server
: The print server to browse. Pass `kPMServerLocal` to specify the local print server. Currently, you may specify only the local print server.

options
: This parameter is reserved for future use. At the present time, pass `NULL`. Passing `NULL` presents the printer browser in the default fashion.

Return Value
A result code. See "Core Printing Result Codes" (page 131). If you specify a server whose printers cannot be browsed, this function returns the error code `kPMInvalidParameter`.

Discussion
This function displays the standard printer browser to allow the user to create a new print queue.

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

Declared In
`PMCore.h`

PMSessionBeginCGDocumentNoDialog

Begins a print job that draws into a Quartz graphics context and suppresses the printing status dialog.

```
OSStatus PMSessionBeginCGDocumentNoDialog (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   PMPageFormat pageFormat
);
```

Parameters

printSession
The printing session that provides a context for the new print job.

printSettings
The print settings to use for the new print job.

pageFormat
The page format to use for the new print job.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
This function starts a print job that draws directly into a Quartz graphics context and should be called within your application's print loop. This function is similar to the function `PMSessionBeginCGDocument` except that the printing status dialog is suppressed.

You must call `PMSessionBeginCGDocumentNoDialog` between the creation and release of a printing session. See the function `PMCreateSession` (page 31). If you present a printing dialog before you call `PMSessionBeginCGDocumentNoDialog`, when calling this function you should use the same `PMPrintSession` (page 110) object you used to present the dialog.

Before you call `PMSessionBeginCGDocumentNoDialog`, you should call `PMSessionValidatePrintSettings` (page 94) and `PMSessionValidatePageFormat` (page 93) to make sure the specified print settings and page format objects are updated and valid. After you call `PMSessionBeginCGDocumentNoDialog`, if you call a function that changes the specified print settings or page format object, the change is ignored for the current print job.

During the print job, the caller cannot obtain a Quickdraw graphics port for the printing session but can only obtain a Quartz graphics context. As a result, this function should be used in conjunction with `PMSessionGetCGGraphicsContext` (page 88) instead of `PMSessionGetGraphicsContext` (page 181).

This function must be called before its corresponding End function (PMSessionEndDocumentNoDialog (page 86)). If the function PMSessionBeginCGDocumentNoDialog returns noErr, you must later call the End function, even if errors occur within the scope of the Begin and End functions.

The printing system automatically handles printing multiple copies. Your application does not need to perform any tasks other than specifying the number of copies in the printing session.

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

Declared In
PMCore.h

PMSessionBeginPageNoDialog

Starts a new page for printing in the specified printing session and suppresses the printing status dialog.

```
OSStatus PMSessionBeginPageNoDialog (
    PMPrintSession printSession,
    PMPageFormat pageFormat,
    const PMRect *pageFrame
);
```

Parameters

printSession

The printing session that provides a context for the print job.

pageFormat

The page format for the new page. If you pass NULL, the printing system uses the page format you passed to PMSessionBeginCGDocumentNoDialog (page 79).

pageFrame

You should pass NULL, as this parameter is currently unsupported.

Return Value
A result code. If the user cancels the print job, this function returns kPMCancel.

Discussion
This function is similar to the function PMSessionBeginPage except that the function PMSessionBeginPageNoDialog suppresses the printing status dialog. You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). You must call the functions PMSessionBeginPageNoDialog and PMSessionEndPageNoDialog (page 86) within the scope of calls to the Begin print job function (PMSessionBeginCGDocumentNoDialog (page 79)) and the End print job function (PMSessionEndDocumentNoDialog (page 86)).

You should call the function PMSessionError (page 87) immediately before you call PMSessionBeginPageNoDialog. If PMSessionError returns an error, then you should not call the function PMSessionBeginPageNoDialog. Because PMSessionBeginPage also initializes the printing graphics context, your application should not make assumptions about the state of the context (for example, the current font) between successive pages. After each call to PMSessionBeginPageNoDialog, your application should call PMSessionGetCGGraphicsContext (page 88) to obtain the current printing context.

If the function PMSessionBeginPageNoDialog returns noErr, you must later call the function PMSessionEndPageNoDialog, even if errors occur within the scope of PMSessionBeginPageNoDialog and PMSessionEndPageNoDialog.

The printing system automatically handles printing multiple copies. Your application does not need to perform any tasks other than specifying the number of copies in the printing session.

Special Considerations

Prior to Mac OS X v10.5, the *pageFormat* parameter is ignored. In Mac OS X v10.5 and later, the printing system supports multiple orientations within a print job. When you call this function and supply a page format, the orientation specified in the page format is used for the current page. Other settings in the page format, such as paper size or scaling, are ignored.

Availability
Available in Mac OS X v10.2 and later.

Declared In
PMCore.h

PMSessionCopyDestinationFormat

Obtains the destination format for a print job.

```
OSStatus PMSessionCopyDestinationFormat (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   CFStringRef *destFormatP
);
```

Parameters

printSession

The printing session that provides a context for the print job.

printSettings

The print settings object for the print job whose destination format you want to obtain.

destFormatP

A pointer to your CFStringRef variable. On return, the variable refers to a Core Foundation string that contains the destination format for the print job. You are responsible for releasing the string. Currently, there are two possible values: kPMDocumentFormatPDF or kPMDocumentFormatPostScript.

If an error occurs, the variable is set to NULL. If the function executes without error and the variable is set to NULL, the print job is set to use the default destination format.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31).

Availability
Available in Mac OS X v10.1 and later.

Declared In
PMCore.h

PMSessionCopyDestinationLocation

Obtains a destination location for a print job.

```
OSStatus PMSessionCopyDestinationLocation (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   CFURLRef *destLocationP
);
```

Parameters

printSession

The printing session that provides a context for the print job.

printSettings

The print settings for the print job whose destination location you want to obtain.

destLocationP

A pointer to your `CFURLRef` variable. On return, the variable refers to a Core Foundation URL that specifies the destination location of the print job. You are responsible for releasing the URL. If `NULL` is returned and the function executes without error (result code is `noErr`), the print job uses the default destination location for the current destination type. If an error occurs, the variable is set to `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Some destination types define a specific kind of destination location for a print job. For example, the destination type `kPMDestinationFile` uses a file system URL to specify where a new file should be created for the print job's output.

Availability

Available in Mac OS X v10.1 and later.

Declared In

`PMCore.h`

PMSessionCopyOutputFormatList

Obtains an array of destination formats supported by the current print destination.

```
OSStatus PMSessionCopyOutputFormatList (
   PMPrintSession printSession,
   PMDestinationType destType,
   CFArrayRef *documentFormatP
);
```

Parameters

printSession

The printing session that provides a context for the print job. The printer associated with this session is queried for the MIME types it supports.

destType

A destination type that specifies the destination for which you want to obtain valid destination formats. See "Destination Types" (page 114) for a list of the possible destination types a print job can have.

documentFormatP

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array that contains a list of destination formats that can be generated for the current print destination. See "Document Format Strings" (page 115) for a list of some of the output formats that can be returned.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.1 and later.

Declared In

`PMCore.h`

PMSessionCreatePageFormatList

Obtains a list of page format objects, each of which describes a paper size available on the specified printer.

```
OSStatus PMSessionCreatePageFormatList (
    PMPrintSession printSession,
    PMPrinter printer,
    CFArrayRef *pageFormatList
);
```

Parameters

printSession

The current printing session.

printer

The printer whose list of page sizes you want to enumerate.

pageFormatList

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array that contains the page format (`PMPageFormat`) objects associated with the specified printer. You are responsible for releasing the array. Each page format object describes a paper size available for the specified printer. If the function fails, then on return the array is `NULL`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

You can use this function to find the available sheet sizes (and the imageable area for them) for a given printer. After you obtain the page format list, you can call the function `PMGetUnadjustedPaperRect` (page 41) for each page format object in the list to obtain the sheet rectangle size. Once you find the paper size you want, call `PMGetUnadjustedPageRect` (page 40) to obtain the imageable area for that paper size.

Availability
Available in Mac OS X v10.1 and later.

Declared In
`PMCore.h`

PMSessionCreatePrinterList

Creates a list of printers available in the specified printing session.

```
OSStatus PMSessionCreatePrinterList (
   PMPrintSession printSession,
   CFArrayRef *printerList,
   CFIndex *currentIndex,
   PMPrinter *currentPrinter
);
```

Parameters

printSession

The printing session whose printer list you want to obtain.

printerList

A pointer to your `CFArrayRef` variable. On return, the variable refers to a Core Foundation array containing a list of printers available in the specified printing session. Each element in the array is a Core Foundation string that contains a printer's name as shown in the user interface. You are responsible for releasing the array.

currentIndex

A pointer to your `CFIndex` variable. On return, the variable contains a value specifying where the current printer is in the printer list.

currentPrinter

A pointer to your `PMPrinter` (page 109) variable. On return, the variable refers to a printer object that represents the current printer. You should not release the printer object without first retaining it. If the printer is the generic printer, the variable is set to `NULL`.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

You can call the function `PMSessionCreatePrinterList` to obtain a valid printer name to pass to the function `PMSessionSetCurrentPrinter`.

Special Considerations

In Mac OS X v10.2 and later, Apple recommends using the function `PMServerCreatePrinterList` (page 78) instead. `PMServerCreatePrinterList` doesn't require a `PMSession` object; it can be called at any time. It also works directly with `PMPrinter` objects.

Availability
Available in Mac OS X v10.1 and later.

See Also
`PMServerCreatePrinterList` (page 78)

Declared In

`PMCore.h`

PMSessionDefaultPageFormat

Assigns default parameter values to a page format object used in the specified printing session.

```
OSStatus PMSessionDefaultPageFormat (
   PMPrintSession printSession,
   PMPageFormat pageFormat
);
```

Parameters

printSession

The printing session for the specified page format object.

pageFormat

The page format object to which you want to assign default values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call the function `PMSessionDefaultPageFormat` between the creation and release of the printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMSessionDefaultPrintSettings

Assigns default parameter values to a print settings object for the specified printing session.

```
OSStatus PMSessionDefaultPrintSettings (
   PMPrintSession printSession,
   PMPrintSettings printSettings
);
```

Parameters

printSession

The printing session for the specified print settings object.

printSettings

The print settings object to which you want to assign default values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call the function `PMSessionDefaultPrintSettings` between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMSessionEndDocumentNoDialog

Ends a print job started by calling the function `PMSessionBeginCGDocumentNoDialog` (page 79) or `PMSessionBeginDocumentNoDialog` (page 179).

```
OSStatus PMSessionEndDocumentNoDialog (
   PMPrintSession printSession
);
```

Parameters

`printSession`

The current printing session. On return, the printing session is no longer valid; however, you must still call the function `PMRelease` (page 76) to release the object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is similar to the function `PMSessionEndDocument` except that the printing status dialog is suppressed.

This function is used to end a print job, and it should be called within your application's print loop after the call to the function `PMSessionEndPageNoDialog` and before releasing the printing session. The same printing session that is created by the function `PMCreateSession` for the Print dialog should be used for the print loop.

The function `PMSessionEndDocumentNoDialog` must be called after its corresponding `Begin` function (`PMSessionBeginCGDocumentNoDialog` (page 79) or `PMSessionBeginDocumentNoDialog` (page 179)). If the `Begin` function returns `noErr`, the function `PMSessionEndDocument` must be called, even if errors occur within the scope of the `Begin` and `End` functions. You should not call `PMSessionEndDocumentNoDialog` if the `Begin` function returns an error.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`PMCore.h`

PMSessionEndPageNoDialog

Indicates the end of drawing the current page for the specified printing session.

```
OSStatus PMSessionEndPageNoDialog (
   PMPrintSession printSession
);
```

Parameters

printSession

The printing session that provides a context for the print job.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is similar to the function PMSessionEndPage except that the printing status dialog is suppressed.

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). You must call the functions PMSessionBeginPageNoDialog and PMSessionEndPageNoDialog within the scope of calls to the Begin print job function (PMSessionBeginCGDocumentNoDialog (page 79)) and the End print job function (PMSessionEndDocumentNoDialog (page 86)).

If the function PMSessionBeginPageNoDialog returns noErr, you must later call the function PMSessionEndPageNoDialog, even if errors occur within the scope of PMSessionBeginPageNoDialog and PMSessionEndPageNoDialog. You should not call PMSessionEndPageNoDialog if PMSessionBeginPageNoDialog returns an error.

Availability

Available in Mac OS X v10.2 and later.

Declared In

PMCore.h

PMSessionError

Obtains the result code for any error returned by the printing session.

```
OSStatus PMSessionError (
   PMPrintSession printSession
);
```

Parameters

printSession

The printing session whose last error you want to obtain.

Return Value

A result code. See "Core Printing Result Codes" (page 131). The constant kPMCancel indicates the user canceled the current print job.

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31).

The PMSessionError function returns the last printing session error, not the last error from a printing function (PMxxx). Because most printing functions return a result code, the PMSessionError function is not required for general error checking. However, you can use PMSessionError in your print loop to determine if the user cancels the current print job or if any other errors occur during printing that are not explicitly returned by one of the other calls. For example, if the user clicks the Cancel button in the status dialog or presses Command-period on the keyboard, this function returns the constant `kPMCancel`. If this or any other error is encountered during the print loop, your application should call the appropriate functions (for example, `PMSessionEndPage` and `PMSessionEndDocument`) to exit the print loop before your application reports the error.

Availability
Available in Mac OS X v10.0 and later.

Related Sample Code
CarbonSketch

Declared In
`PMCore.h`

PMSessionGetCGGraphicsContext

Obtains the Quartz graphics context for the current page in a printing session.

```
OSStatus PMSessionGetCGGraphicsContext (
   PMPrintSession printSession,
   CGContextRef *context
);
```

Parameters

printSession
The printing session whose Quartz graphics context you want to obtain.

context
A pointer to your `CGContextRef` variable. On return, the variable refers to the Quartz graphics context for the current page in the specified printing session. The context's origin is at the lower-left corner of the sheet of paper, not the imageable area. You should not release the context without first retaining it. The context is valid only for the current page; you should not retain it beyond the end of the page.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
If you're using Quartz 2D to draw the content for a print job, after each call to `PMSessionBeginPage` you should call `PMSessionGetCGGraphicsContext` to obtain the Quartz graphics context for the current page. Note that before you can use the function `PMSessionGetCGGraphicsContext`, you must have called `PMSessionBeginCGDocument` or `PMSessionBeginCGDocumentNoDialog` (page 79) instead of `PMSessionBeginDocument` or `PMSessionBeginDocumentNoDialog` (page 179).

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

Declared In
`PMCore.h`

PMSessionGetCurrentPrinter

Obtains the current printer associated with a printing session.

```
OSStatus PMSessionGetCurrentPrinter (
   PMPrintSession printSession,
   PMPrinter *currentPrinter
);
```

Parameters

printSession

The printing session whose printer you want to obtain.

currentPrinter

A pointer to your `PMPrinter` (page 109) variable. On return, the variable refers to the printer associated with the specified printing session. The printer object is valid as long as the printing session is valid or the current printer hasn't changed. You should not release this object without first retaining it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMSessionSetCurrentPMPrinter` (page 91)

Declared In

`PMCore.h`

PMSessionGetDataFromSession

Obtains application-specific data previously stored in a printing session object.

```
OSStatus PMSessionGetDataFromSession (
   PMPrintSession printSession,
   CFStringRef key,
   CFTypeRef *data
);
```

Parameters

printSession

The printing session whose data you want to obtain.

key

The key that uniquely identifies the data to be retrieved. You specify this key when you store the data using the function `PMSessionSetDataInSession` (page 91).

data

A pointer to your `CFTypeRef` variable. On return, the variable refers to the data retrieved from the printing session.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMSessionSetDataInSession` (page 91)

Declared In

`PMCore.h`

PMSessionGetDestinationType

Obtains the output destination for a print job.

```
OSStatus PMSessionGetDestinationType (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   PMDestinationType *destTypeP
);
```

Parameters

printSession

The printing session that provides a context for the print job. This must be the same printing session used for the Print dialog. The printing session contains the preview setting, which can override the destination type in the print settings.

printSettings

The print settings for the print job whose destination you want to obtain.

destTypeP

A pointer to your `PMDestinationType` variable. On return, the variable contains the destination type for the specified print job. Possible values include:

- `kPMDestinationPrinter` (output to a printer)
- `kPMDestinationFile` (output to a file)
- `kPMDestinationFax` (output to a fax)
- `kPMDestinationPreview` (output to print preview)
- `kPMDestinationProcessPDF` (output to a PDF workflow option)

See "Destination Types" (page 114) for a complete description of the destination type constants.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

All of the destination types are stored in the print settings object except for `kPMDestinationPreview`, which is stored in the printing session object. If the destination type is set as preview, the preview setting overrides the destination set in the print settings object.

Availability
Available in Mac OS X v10.1 and later.

Declared In
`PMCore.h`

PMSessionSetCurrentPMPrinter

Changes the current printer for a printing session.

```
OSStatus PMSessionSetCurrentPMPrinter (
   PMPrintSession session,
   PMPrinter printer
);
```

Parameters

`session`

The printing session whose printer you want to change.

`printer`

The new printer for the printing session.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability
Available in Mac OS X v10.3 and later.

See Also
`PMSessionGetCurrentPrinter` (page 88)

Declared In
`PMCore.h`

PMSessionSetDataInSession

Stores your application-specific data in a printing session object.

```
OSStatus PMSessionSetDataInSession (
   PMPrintSession printSession,
   CFStringRef key,
   CFTypeRef data
);
```

Parameters

`printSession`

The printing session in which you want to store application-specific data.

`key`

A key that uniquely identifies the data being added. This key is required to retrieve the data using the function `PMSessionGetDataFromSession` (page 89).

data

The data to be stored in the printing session.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMSessionGetDataFromSession` (page 89)

Declared In

`PMCore.h`

PMSessionSetDestination

Sets the destination location, format, and type for a print job.

```
OSStatus PMSessionSetDestination (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   PMDestinationType destType,
   CFStringRef destFormat,
   CFURLRef destLocation
);
```

Parameters

printSession

The printing session that provides a context for the print job.

printSettings

The print settings for the print job whose destination you want to set.

destType

The destination type for the print job associated with the specified printing session and print settings. Possible values include:

- `kPMDestinationPrinter` (output to a printer)
- `kPMDestinationFile` (output to a file)
- `kPMDestinationFax` (output to a fax)
- `kPMDestinationPreview` (output to print preview)
- `kPMDestinationProcessPDF` (output to a PDF workflow option)

See "Destination Types" (page 114) for a complete description of destination types you can specify.

destFormat

The MIME type to be generated for the specified destination type. Pass `NULL` if you want to use the default format for the specified destination type. To obtain a list of valid formats for a given destination type, use the function `PMSessionCopyOutputFormatList` (page 82).

destLocation

A reference to a Core Foundation URL that specifies a destination location. You can provide this if the destination type supports a destination location. Otherwise, pass `NULL`. For example, if the destination type is a file (`kPMDestinationFile`) you can supply a file system URL to specify where the file resides.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can use the function `PMSessionSetDestination` when you want to send print output to a file without requiring user interaction. You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.1 and later.

Declared In

`PMCore.h`

PMSessionSetError

Sets the value of the current result code for the specified printing session.

```
OSStatus PMSessionSetError (
   PMPrintSession printSession,
   OSStatus printError
);
```

Parameters

printSession

The printing session whose result code you want to set.

printError

The result code you want to set. This result code is returned by the `PMSessionError` function.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

You can use this function to terminate a printing session if your application encounters any errors inside the print loop. Typically, this function is used by an application's idle function. The idle function isn't called in Mac OS X, so this usage is not available.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMCore.h`

PMSessionValidatePageFormat

Updates the values in a page format object and validates them against the current formatting printer.

```
OSStatus PMSessionValidatePageFormat (
   PMPrintSession printSession,
   PMPageFormat pageFormat,
   Boolean *result
);
```

Parameters

printSession

The printing session for the specified page format object.

pageFormat

The page format object to validate.

result

A pointer to your Boolean variable. On return, `true` if the function set the page format object to default values; otherwise, `false`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of the printing session. See the function `PMCreateSession` (page 31).

The function `PMSessionValidatePageFormat` validates the page format object against the current formatting printer. The formatting printer is displayed in the Format for pop-up menu in the Page Setup dialog. The default formatting printer is the generic Any Printer. If the page format object contains values that are not valid for the formatting printer, the page format object is set to default values and the *result* parameter is set to `true`.

Validating a page format object also causes calculated fields (such as the adjusted paper and page rectangles) to be updated based on the changed settings (such as resolution, scaling, and page orientation). If the page format object contains values that are valid for the formatting printer but need to be updated, the *result* parameter is set to `false`.

After you call any function that makes changes to a page format object (such as `PMSetOrientation`), you should call the function `PMSessionValidatePageFormat` to validate the page format object before using that object.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMSessionValidatePrintSettings

Validates a print settings object within the context of the specified printing session.

```
OSStatus PMSessionValidatePrintSettings (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   Boolean *result
);
```

Parameters

printSession

The printing session for the specified print settings object.

printSettings

The print settings object to validate.

result

A pointer to your Boolean variable. On return, `true` if any parameters changed, or `false` if no parameters changed.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMSetCollate

Specifies whether the job collate option is selected.

```
OSStatus PMSetCollate (
   PMPrintSettings printSettings,
   Boolean collate
);
```

Parameters

printSettings

The print settings object whose job collate option you want to set.

collate

If `true`, the job collate option is selected; if `false` the option is not selected.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The Collated checkbox is displayed in the Copies & Pages pane of the Print dialog. This option determines how printed material is organized. For example, if you have a document that is three pages long and you are printing multiple copies with the Collated option selected, the job prints pages 1, 2, and 3 in that order and then repeats. However, if the Collated option is not selected and you're printing multiple copies of those same three pages, the job prints copies of page 1, then copies of page 2, and finally copies of page 3.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.2 and later.

See Also

`PMGetCollate` (page 33)

Declared In

`PMCore.h`

PMSetCopies

Sets the initial value for the number of copies to be printed.

```
OSStatus PMSetCopies (
   PMPrintSettings printSettings,
   UInt32 copies,
   Boolean lock
);
```

Parameters

`printSettings`

The print settings object you want to initialize.

`copies`

The initial value of the number of copies to print.

`lock`

The lock state of the setting. Locking is not supported at this time.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMGetCopies` (page 33)

Declared In

`PMCore.h`

PMSetDuplex

Sets the duplex mode.

```
OSStatus PMSetDuplex (
   PMPrintSettings printSettings,
   PMDuplexMode duplexSetting
);
```

Parameters

printSettings

The print settings object whose duplex mode you want to set.

duplexSetting

The new duplex mode setting. Possible values include:

- `kPMDuplexNone` (one-sided printing)
- `kPMDuplexNoTumble` (two-sided printing)
- `kPMDuplexTumble` (two-sided printing with tumbling)

See "Duplex Modes" (page 115) for a full description of the constants you can use to specify the new setting.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Duplex printing is a print job that prints on both sides of the paper. Two-Sided printing controls are displayed in the Layout pane of the Print dialog. Note that not all printers support duplex printing. This function specifies a setting that might not be available on a given destination.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.4 and later.

See Also

`PMGetDuplex` (page 34)

Declared In

`PMCore.h`

PMSetFirstPage

Sets the default page number of the first page to be printed.

```
OSStatus PMSetFirstPage (
   PMPrintSettings printSettings,
   UInt32 first,
   Boolean lock
);
```

Parameters

printSettings

The print settings object whose first page number you want to set.

first
: The page number of the first page to print. This value appears in the From field of the Print dialog.

lock
: The lock state of the setting. Locking is not supported at this time.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Typically, this function isn't used. In Mac OS X, if you call the function `PMSetPageRange` (page 101) and then call `PMSetFirstPage` or `PMSetLastPage` using the same page range you specified for `PMSetPageRange`, then the Print dialog shows the From button selected. If you use the constant `kPMPrintAllPages` to set the page range with the function `PMSetPageRange`, then the Print dialog opens with the All button selected regardless of whether you also call `PMSetFirstPage` or `PMSetLastPage`.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMGetFirstPage` (page 34)

Related Sample Code

CarbonSketch

Declared In

`PMCore.h`

PMSetLastPage

Sets the page number of the last page to be printed.

```
OSStatus PMSetLastPage (
   PMPrintSettings printSettings,
   UInt32 last,
   Boolean lock
);
```

Parameters

printSettings
: The print settings object whose last page number you want to set.

last
: The page number of the last page to print. This value appears in the To field of the Print dialog. Pass the constant `kPMPrintAllPages` to print the entire document.

lock
: The lock state of the setting. Locking is not supported at this time.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion
Typically, you call this function after the Print dialog is displayed to indicate the number of the last page number to be printed. In Mac OS X, setting the last page provides information used by the progress dialog that is shown during printing.

If you call the function `PMSetPageRange` (page 101) and then call `PMSetFirstPage` or `PMSetLastPage` using the same page range you specified for `PMSetPageRange`, then the Print dialog shows the From button selected. If you use the constant `kPMPrintAllPages` to set the page range with the function `PMSetPageRange`, then the Print dialog opens with the All button selected regardless of whether you also call `PMSetFirstPage` or `PMSetLastPage`.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability
Available in Mac OS X v10.0 and later.

See Also
`PMGetLastPage` (page 35)

Related Sample Code
CarbonSketch

Declared In
`PMCore.h`

PMSetOrientation

Sets the page orientation for printing.

```
OSStatus PMSetOrientation (
   PMPageFormat pageFormat,
   PMOrientation orientation,
   Boolean lock
);
```

Parameters

`pageFormat`

The page format object whose page orientation you want to set.

`orientation`

A constant specifying the desired page orientation. Supported values are:

- `kPMPortrait`
- `kPMLandscape`
- `kPMReversePortrait` (Mac OS X v10.5 and later)
- `kPMReverseLandscape`

See "Page Orientation Constants" (page 117) for a full description of the values you can use to specify page orientation.

`lock`

The lock state of the setting. You should pass `kPMUnlocked`. Locking is not supported at this time.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Special Considerations

In Mac OS X 10.4 and earlier, if you want to set the page orientation you need to call this function before initiating the print job (for example, by calling `PMSessionBeginCGDocument`). The page orientation you set applies to the entire print job. In Mac OS X 10.5 and later, you can use this function to change the orientation of an individual page in a print job by passing the updated page format to `PMSessionBeginPage` or `PMSessionBeginPageNoDialog`.

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMGetOrientation` (page 36)

Declared In

`PMCore.h`

PMSetPageFormatExtendedData

Stores your application-specific data in a page format object.

```
OSStatus PMSetPageFormatExtendedData (
   PMPageFormat pageFormat,
   OSType dataID,
   UInt32 size,
   void *extendedData
);
```

Parameters

pageFormat

The page format object in which to store your extended data.

dataID

A 4-character code that identifies your data. This is typically your application's creator code. If your creator code is outside the ASCII 7-bit character range 0x20–0x7F, you need to use a different 4-character code.

size

The size, in bytes, of the data to be stored in the page format object.

extendedData

A pointer to the application-specific data you want to store in the page format object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can retrieve the data you store with the function `PMSetPageFormatExtendedData` by calling the function `PMGetPageFormatExtendedData`.

Availability

Available in Mac OS X v10.0 and later.

See Also

`PMGetPageFormatExtendedData` (page 37)

Declared In

`PMCore.h`

PMSetPageRange

Sets the valid range of pages that can be printed.

```
OSStatus PMSetPageRange (
   PMPrintSettings printSettings,
   UInt32 minPage,
   UInt32 maxPage
);
```

Parameters

printSettings

The print settings object whose page range you want to set.

minPage

The minimum page number allowed. This value appears as the default in the From field of the Print dialog.

maxPage

The maximum page number allowed. This value appears as the default in the To field of the Print dialog. Pass the constant `kPMPrintAllPages` to allow the user to print the entire document. If the first page is set to 1, then passing `kPMPrintAllPages` as the maximum page number causes the All button to be selected.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The function `PMSetPageRange` allows applications to set the minimum and maximum page numbers that can be printed for a document. If the user enters a value outside of this range in the Print dialog, the value is set to the closest allowed value. You can use the `PMGetFirstPage` (page 34) and `PMGetLastPage` (page 35) functions to obtain the values entered by the user in the Print dialog.)

If you call the function `PMSetPageRange` to set the maximum page to a value other than the constant `kPMPrintAllPages`, the function `PMSetPageRange` causes the page range in the Print dialog to be properly restricted to the specified range. If you call the function `PMSetPageRange` without also calling the functions `PMSetFirstPage` or `PMSetLastPage`, then the Print dialog shows the specified page range in the From and To fields but with the All button selected. If you call the function `PMSetPageRange` and then call `PMSetFirstPage` or `PMSetLastPage` using the same page range you specified for `PMSetPageRange`, then the Print dialog shows the From button selected.

In all cases, if your application sets a range with `PMSetPageRange` and subsequently calls `PMSetFirstPage` (page 97) or `PMSetLastPage` (page 98) with values outside of the specified range, Core Printing returns a result code of `kPMValueOutOfRange`. Conversely, if your application calls `PMSetPageRange` after calling `PMSetFirstPage` or `PMSetLastPage` (or after displaying the Print dialog), the page range specified by `PMSetPageRange` takes precedence, and the first and last page values are adjusted accordingly.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

See Also

PMGetPageRange (page 38)

Related Sample Code

CarbonSketch

Declared In

PMCore.h

PMSetPrintSettingsExtendedData

Stores your application-specific data in a print settings object.

```
OSStatus PMSetPrintSettingsExtendedData (
   PMPrintSettings printSettings,
   OSType dataID,
   UInt32 size,
   void *extendedData
);
```

Parameters

printSettings

The print settings object in which to store your application-specific data.

dataID

A 4-character code that will be used to identify your data. The 4-character code must not contain any characters outside the standard ASCII 7-bit character range 0x20–0x7F. This is typically your application's creator code.

size

The size, in bytes, of the data to be stored in the print settings object.

extendedData

A pointer to a buffer that contains the extended data you want to store.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can retrieve the data you store with the function PMSetPrintSettingsExtendedData by calling the function PMGetPrintSettingsExtendedData.

You may find it easier to use the functions PMPrintSettingsSetValue (page 74) and PMPrintSettingsGetValue (page 73) to store and retrieve user-defined data in a print settings object. If you use these functions, make sure that the custom keys you define for your private data do not conflict with other print settings keys.

Availability

Available in Mac OS X v10.0 and later.

See Also

PMGetPrintSettingsExtendedData (page 39)

Declared In

PMCore.h

PMSetScale

Sets the scaling factor for the page and paper rectangles.

```
OSStatus PMSetScale (
   PMPageFormat pageFormat,
   double scale
);
```

Parameters

pageFormat

The page format object whose scaling factor you want to set.

scale

The desired scaling factor expressed as a percentage. For example, for 50 percent scaling, pass a value of 50.0; for no scaling, pass 100.0.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can call the function PMSetScale to change the scaling factor that appears when your application invokes the Page Setup dialog.

If you call PMSetScale after calling PMSessionPageSetupDialog, make sure you call PMSessionValidatePageFormat (page 93) before you call PMSessionBeginCGDocument or PMSessionBeginDocument.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

See Also

PMGetScale (page 40)

Declared In

PMCore.h

PMWorkflowCopyItems

Obtains an array of the available PDF workflow items.

```
OSStatus PMWorkflowCopyItems (
   CFArrayRef *workflowItems
);
```

Parameters

workflowItems

A pointer to your CFArrayRef variable. On return, the variable refers to an Core Foundation array. Each element in the array is a dictionary that describes either a PDF workflow item or a folder containing a set of PDF workflow items. For a list of possible keys, see "PDF Workflow Dictionary Keys" (page 117). You are responsible for releasing the array.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMCore.h`

PMWorkflowSubmitPDFWithOptions

Submits a PDF file for workflow processing using the specified CUPS options string.

```
OSStatus PMWorkflowSubmitPDFWithOptions (
   CFURLRef workflowItem,
   CFStringRef title,
   const char *options,
   CFURLRef pdfFile
);
```

Parameters

workflowItem

A file system URL pointing to the workflow item that will handle the PDF file. See `PMWorkflowCopyItems` (page 103). The following table describes the different types of workflow items for this function.

Workflow item	Description
Automator action	The action is executed for the PDF file. Available in Mac OS X v10.4 and later.
Folder alias	The PDF file is moved to the resolved folder.
Application or application alias	The application is sent an open event along with a reference to the PDF file.
Compiled AppleScript	The script is run with an open event along with a reference to the PDF file.
Executable tool	The tool is run with the following parameters: *title*, *options*, and *pdfFile*.

title

The user-displayable name of the PDF document.

options

A string of CUPS-style key-value pairs that may be passed to the PDF workflow item. This parameter can be `NULL` in which case an empty string of options is used.

pdfFile

A file system URL pointing to the PDF file to be processed by the workflow item.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
The printing system uses this function in conjunction with the function `PMWorkflowCopyItems` (page 103) to implement the PDF workflow button in the Print dialog.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMCore.h`

PMWorkflowSubmitPDFWithSettings

Submits a PDF file for workflow processing using the specified print settings.

```
OSStatus PMWorkflowSubmitPDFWithSettings (
   CFURLRef workflowItem,
   PMPrintSettings settings,
   CFURLRef pdfFile
);
```

Parameters

workflowItem

A file system URL pointing to the workflow item that will handle the PDF file. See `PMWorkflowCopyItems` (page 103). The following table describes the different types of workflow items for this function.

Workflow item	Description
Automator action	The action is executed for the PDF file. Available in Mac OS X v10.4 and later.
Folder alias	The PDF file is moved to the resolved folder.
Application or application alias	The application is sent an open event along with a reference to the PDF file.
Compiled AppleScript	The script is run with an open event along with a reference to the PDF file.
Executable tool	The tool is run with the specified settings and PDF file. This function converts these parameters into a CUPS options string and passes the options string to the tool.

settings

The print settings to apply to the PDF document. These settings are passed to the workflow item as a CUPS options string.

pdfFile

A file system URL pointing to the PDF file to be processed by the workflow item.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
The printing system uses this function in conjunction with the function `PMWorkflowCopyItems` (page 103) to implement the PDF workflow button in the Print dialog.

Special Considerations

In Mac OS X v10.4 and earlier, this function is not implemented and returns an error. You can use the function `PMWorkflowSubmitPDFWithOptions` (page 104) together with the function `PMPrintSettingsToOptions` (page 75) instead.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`PMCore.h`

Callbacks

PMIdleProcPtr

Defines a pointer to an idle function. (**Deprecated.** There is no replacement; this callback function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
typedef void (*PMIdleProcPtr) (void);
```

You would declare your idle function like this if you were to name it `MyPrintIdleCallback`:

```
void MyPrintIdleCallback (void);
```

Discussion

If you install an idle function using the function `PMSessionSetIdleProc`, the printing system calls your idle function periodically during your print loop. Your idle function can display application status while printing, but it should not duplicate information displayed by the printing system or the printer driver. If you don't install an idle function, you get the standard dialog for the current driver in Mac OS 8 and 9.

Your idle function must check whether the user has pressed Command-period, in which case your application should stop its printing operation. If your status dialog contains a button to cancel the printing operation, your idle function should also check for clicks in the button and respond accordingly.

To provide a pointer to your idle function, you create a universal procedure pointer (UPP) of type `PMIdleUPP`, using the function `NewPMIdleUPP`. You can do so with code similar to the following:

```
PMIdleUPP MyPrintIdleUPP;
MyPrintIdleUPP = NewPMIdleUPP (&MyPrintIdleCallback);
```

When your print job is completed, you should use the function `DisposePMIdleUPP` to dispose of the universal procedure pointer associated with your idle function. However, if you will use the same idle function in subsequent print jobs, you can reuse the same UPP, rather than dispose of it and later create a new UPP.

Special Considerations

Your idle function is not called in Mac OS X. It's only called in Mac OS 8 and 9.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
PMCoreDeprecated.h

Data Types

PMDialog

An opaque type that represents a custom printing dialog.

```
typedef struct OpaquePMDialog* PMDialog;
```

Discussion
This data type is used by functions that are not recommended or deprecated.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
PMDefinitionsDeprecated.h

PMIdleUPP

A type that defines a universal procedure pointer to an idle callback.

```
typedef PMIdleProcPtr PMIdleUPP;
```

Discussion
This data type is used by functions that are not recommended or deprecated.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
PMCoreDeprecated.h

PMLanguageInfo

A data structure that contains level, version, and release information for the imaging language used by a printer driver.

```
struct PMLanguageInfo {
    Str32 level;
    Str32 version;
    Str32 release;
};
```

Fields

`level`

Specifies the level of the imaging language used by the printer driver.

`version`

Specifies the version of the imaging language.

`release`

Specifies the release of the imaging language.

PMObject

The base type for all the opaque types used in Core Printing.

```
typedef const void* PMObject;
```

Discussion

`PMObject` is the base type for opaque types such as `PMPrintSession`, `PMPageFormat`, `PMPrintSettings`, `PMPrinter`, `PMPaper`, `PMPreset`, and `PMServer`. `PMObject` is used in functions such as `PMRetain` (page 77) and `PMRelease` (page 76) that operate on any opaque type.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMDefinitions.h`

PMPageFormat

An opaque type that stores the settings in the Page Setup dialog.

```
typedef struct OpaquePMPageFormat* PMPageFormat;
```

Discussion

Your application uses page format objects to store information such as the paper size, orientation, and scale of pages in a printing session. To create a page format object, you use the function `PMCreatePageFormat` (page 29). A new page format object is empty and unusable until you call `PMSessionDefaultPageFormat` (page 85) or `PMCopyPageFormat` (page 27) to initialize the settings. You can also use the functions `PMSetPageFormatExtendedData` (page 100) and `PMGetPageFormatExtendedData` (page 37) to store and retrieve application-specific data in a page format object.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PMDefinitions.h`

PMPaper

An opaque type that stores information about the paper used in a print job.

```
typedef struct OpaquePMPaper* PMPaper;
```

Discussion
Your application uses paper objects to identify standard and custom types of printing paper.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMDefinitions.h`

PMPaperMargins

A data structure that specifies the unprintable area of a paper object.

```
typedef PMRect PMPaperMargins;
```

Discussion
Your application specifies paper margins when calling the function `PMPaperCreateCustom` (page 43) to create a custom paper type. You can obtain a paper's margins with the function `PMPaperGetMargins` (page 46).

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMDefinitions.h`

PMPreset

An opaque type that stores information about a named preset available for a print job.

```
typedef struct OpaquePMPreset* PMPreset;
```

Discussion
Your application uses a preset object to identify a named preset in the Print dialog. You typically obtain an instance of this type using the function `PMPrinterCopyPresets` (page 53).

Availability
Available in Mac OS X v10.3 and later.

Declared In
`PMDefinitions.h`

PMPrinter

An opaque type that represents a printer.

```
typedef struct OpaquePMPrinter* PMPrinter;
```

Discussion
You typically obtain a printer object using the function `PMSessionGetCurrentPrinter` (page 88) or `PMServerCreatePrinterList` (page 78).

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMDefinitions.h`

PMPrintSession

An opaque type that stores information about a print job.

```
typedef struct OpaquePMPrintSession* PMPrintSession;
```

Discussion
A printing session object contains information that's needed by the page format and print settings objects, such as default page format and print settings values. For this reason, some printing functions can be called only after you have created a printing session object. For example, setting defaults for or validating page format and print settings objects can only be done after you have created a printing session object. Your application creates a printing session object using the function `PMCreateSession` (page 31).

You can use a printing session to implement multithreaded printing, and you can create multiple sessions within a single-threaded application. If your application does not use sheets, then your application can open only one dialog at a time. Each printing session can have its own dialog, and settings changed in one dialog are independent of settings in any other dialog.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`PMDefinitions.h`

PMPrintSettings

An opaque type that stores the settings in the Print dialog.

```
typedef struct OpaquePMPrintSettings* PMPrintSettings;
```

Discussion
Your application uses print settings objects to store information such as the number of copies and the range of pages to print in a printing session. To create a print settings object, you use the function `PMCreatePrintSettings` (page 30). A new print settings object is empty and unusable until you call `PMSessionDefaultPrintSettings` (page 85) or `PMCopyPrintSettings` (page 28) to initialize the settings. You can also use the functions `PMSetPrintSettingsExtendedData` (page 102) and `PMGetPrintSettingsExtendedData` (page 39) to store and retrieve application-specific data in a print settings object.

Availability
Available in Mac OS X v10.0 and later.

Declared In
PMDefinitions.h

PMRect

A data structure that describes a rectangle using four double-precision coordinates.

```
struct PMRect {
    double top;
    double left;
    double bottom;
    double right;
};
```

Fields

top

The vertical coordinate for the upper-left point of the rectangle.

left

The horizontal coordinate for the upper-left point of the rectangle.

bottom

The vertical coordinate for the lower-right point of the rectangle.

right

The horizontal coordinate for the lower-right point of the rectangle.

PMResolution

A data structure that contains printing resolution information.

```
struct PMResolution {
    double hRes;
    double vRes;
};
```

Fields

hRes

The horizontal resolution in dots per inch (dpi).

vRes

The vertical resolution in dots per inch (dpi).

Discussion

The functions PMGetResolution (page 176) and PMPrinterGetPrinterResolution (page 178) use this structure to return printing resolution information. Your application can pass this information to the function PMSetResolution (page 185).

PMServer

An opaque type that identifies a local or remote print server.

```
typedef struct OpaquePMServer* PMServer;
```

Availability
Available in Mac OS X v10.2 and later.

Declared In
`PMDefinitions.h`

PMPrintContext

An opaque type that describes the graphics environment for printing a document.

```
typedef struct OpaquePMPrintContext* PMPrintContext;
```

Discussion
This data type is used by functions that are not recommended or deprecated.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`PMDefinitionsDeprecated.h`

PMColorMode

A type that specifies color modes to use for printing.

```
typedef UInt16 PMColorMode;
```

Discussion
This data type is used by functions that are not recommended or deprecated.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`PMDefinitionsDeprecated.h`

Constants

Data Not Wanted Constants

Constants your application can use to indicate it does not need certain types of data returned by various printing functions.

```
#define kPMNoData  NULL
#define kPMDontWantSize  NULL
#define kPMDontWantData  NULL
#define kPMDontWantBoolean  NULL
#define kPMNoPrintSettings  NULL
#define kPMNoPageFormat  NULL
#define kPMNoReference  NULL
```

Constants

`kPMNoData`

Specifies that your application does not need data returned for a particular parameter. For future compatibility, you are encouraged to use one of the following constants in cases where a specific type of data is not required.

`kPMDontWantSize`

Specifies that your application does not need the size information returned by the printing function.

`kPMDontWantData`

Specifies that your application does not need the data returned by the printing function.

`kPMDontWantBoolean`

Specifies that your application does not need a Boolean value returned by the printing function.

`kPMNoPrintSettings`

Specifies that your application does not need a `PMPrintSettings` object returned by the printing function.

`kPMNoPageFormat`

Specifies that your application does not need a `PMPageFormat` object returned by the printing function.

`kPMNoReference`

Specifies that your application does not need a reference returned by the printing function.

Data Representation Formats

Constants that specify the format of the data representation created with the functions `PMPageFormatCreateDataRepresentation` (page 41) and `PMPrintSettingsCreateDataRepresentation` (page 71).

```
enum PMDataFormat {
  kPMDataFormatXMLDefault = 0,
  kPMDataFormatXMLMinimal = 1,
  kPMDataFormatXMLCompressed = 2
};
typedef enum PMDataFormat PMDataFormat;
```

Constants

`kPMDataFormatXMLDefault`

Specifies a data format that is compatible with all Mac OS X versions. Data in this format can be used with the `PMUnflattenXXX` functions present in versions of Mac OS X prior to 10.5. This format is a pure XML representation of the data. However, this format is much larger than the more modern data formats described below.

Available in Mac OS X v10.5 and later.

Declared in `PMDefinitions.h`.

`kPMDataFormatXMLMinimal`

Specifies an uncompressed data format that is approximately 3-5 times smaller than `kPMDataFormatXMLDefault`. This data format is only compatible with Mac OS X v10.5 and later. This format is a good choice when you do not need to use the data in versions of Mac OS X prior to 10.5 and you need a pure XML representation of the data.

Available in Mac OS X v10.5 and later.

Declared in `PMDefinitions.h`.

`kPMDataFormatXMLCompressed`

Specifies a compressed data format that is approximately 20 times smaller than `kPMDataFormatXMLDefault`. This data format is only compatible with Mac OS X v10.5 and later. This format is a good choice when you do not need to use the data in versions of Mac OS X prior to 10.5 and the minimum data size is important. Note that this format is not a pure XML representation of the data.

Available in Mac OS X v10.5 and later.

Declared in `PMDefinitions.h`.

Destination Types

Constants that specify a destination for a print job.

```
typedef UInt16 PMDestinationType;
enum {
    kPMDestinationInvalid = 0,
    kPMDestinationPrinter = 1,
    kPMDestinationFile = 2,
    kPMDestinationFax = 3,
    kPMDestinationPreview = 4,
    kPMDestinationProcessPDF = 5
};
```

Constants

`kPMDestinationInvalid`

Specifies the destination is invalid.

Available in Mac OS X v10.1 and later.

Declared in `PMDefinitions.h`.

`kPMDestinationPrinter`

Specifies output to a printer.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMDestinationFile`

Specifies output to a file.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMDestinationFax`

Specifies output to a fax. This destination is currently not supported.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMDestinationPreview`
Specifies output to print preview.

Available in Mac OS X v10.1 and later.

Declared in `PMDefinitions.h`.

`kPMDestinationProcessPDF`
Specifies output to a PDF workflow option.

Available in Mac OS X v10.4 and later.

Declared in `PMDefinitions.h`.

Document Format Strings

Constants that specify the document format for a print job.

```
#define kPMDocumentFormatDefault
            CFSTR("com.apple.documentformat.default")
#define kPMDocumentFormatPDF
            CFSTR("application/pdf")
#define kPMDocumentFormatPICT
            CFSTR("application/vnd.apple.printing-pict")
#define kPMDocumentFormatPICTPS
            CFSTR("application/vnd.apple.printing-pict-ps")
#define kPMDocumentFormatPostScript
            CFSTR("application/postscript")
```

Constants

`kPMDocumentFormatDefault`
Specifies the default format for the printing system. In Mac OS X, the default format is PDF.

`kPMDocumentFormatPDF`
Specifies PDF.

`kPMDocumentFormatPICT`
Specifies PICT format.

`kPMDocumentFormatPICTPS`
Specifies PICT format with embedded PostScript.

`kPMDocumentFormatPostScript`
Specifies PostScript format.

Duplex Modes

Constants that specify duplex mode settings.

```
typedef UInt32 PMDuplexMode;
enum {
    kPMDuplexNone = 0x0001,
    kPMDuplexNoTumble = 0x0002,
    kPMDuplexTumble = 0x0003,
    kPMSimplexTumble = 0x0004
};
```

Constants

`kPMDuplexNone`

Print on only one side of the paper.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMDuplexNoTumble`

Print on both sides of the paper, with both sides oriented in the same direction (no tumbling.)

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMDuplexTumble`

Print on both sides of the paper, with the output on the second side flipped relative to the first side (tumbling on.)

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMSimplexTumble`

Print on only one side of the paper, but tumble the images while printing. This mode is not supported at this time.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

Graphics Context Types

Constants that specify the graphics context for a print job.

```
#define kPMGraphicsContextDefault
        CFSTR("com.apple.graphicscontext.default")
#define kPMGraphicsContextQuickdraw
        CFSTR("com.apple.graphicscontext.quickdraw")
#define kPMGraphicsContextCoreGraphics
        CFSTR("com.apple.graphicscontext.coregraphics")
```

Constants

`kPMGraphicsContextDefault`

Specifies the default graphics context for the application's runtime environment.

`kPMGraphicsContextQuickdraw`

Specifies a QuickDraw graphics context.

`kPMGraphicsContextCoreGraphics`

Specifies a Quartz graphics context. The default coordinate system for a Quartz printing context is not the same as that used for a QuickDraw printing context. A Quartz printing context defines the coordinates of the lower-left corner of the paper as (0,0) whereas the origin for a QuickDraw is at the upper-right corner of the paper's imageable area.

Page Orientation Constants

Constants that specify page orientation.

```
typedef UInt16 PMOrientation;
enum {
    kPMPortrait = 1,
    kPMLandscape = 2,
    kPMReversePortrait = 3,
    kPMReverseLandscape = 4
};
```

Constants

`kPMPortrait`

Specifies portrait (vertical) page orientation. Portrait orientation performs no alteration of the logical page.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMLandscape`

Specifies landscape (horizontal) orientation. Landscape orientation performs a 90° counterclockwise rotation on the logical page image and sets the Quartz origin to the upper-left corner of the unrotated logical page with positive y-values increasing across and to the right of the unrotated page. This has the effect of rotating the logical page image 90° clockwise. In other words, the image appears on the unrotated page as if it were rotated 90° clockwise.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMReversePortrait`

Specifies reverse portrait orientation. Reverse portrait orientation performs a 180° rotation on the logical page rectangle and sets the Quartz origin to the upper-right corner of the unrotated logical page with positive y-values increasing downwards. This has the effect of rotating the logical page image 180°. Reverse portrait orientation is supported in Mac OS X v10.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMReverseLandscape`

Specifies reverse landscape page orientation. Landscape orientation performs a 90° clockwise rotation on the logical page rectangle and sets the Quartz origin to the lower-right corner of the unrotated logical page, with the positive y-values increasing to across and to the left of the unrotated page. This has the effect of rotating the logical page image 90° counterclockwise. In other words, the image appears on the unrotated page as if it were rotated 90° counterclockwise.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

PDF Workflow Dictionary Keys

Constants that specify the keys in a PDF workflow dictionary.

```
#define kPDFWorkFlowItemURLKey CFSTR("itemURL")
#define kPDFWorkflowDisplayNameKey CFSTR("displayName")
#define kPDFWorkflowFolderURLKey CFSTR("folderURL")
#define kPDFWorkflowItemsKey CFSTR("items")
```

Constants

`kPDFWorkFlowItemURLKey`

The URL to the PDF workflow item.

Available in Mac OS X v10.3 and later.

Declared in `PMDefinitions.h`.

`kPDFWorkflowDisplayNameKey`

The user-displayable name for the PDF workflow item.

Available in Mac OS X v10.3 and later.

Declared in `PMDefinitions.h`.

`kPDFWorkflowFolderURLKey`

The URL to the folder containing PDF workflow items.

Available in Mac OS X v10.5 and later.

Declared in `PMDefinitions.h`.

`kPDFWorkflowItemsKey`

A Core Foundation array describing the PDF workflow items in the folder.

Available in Mac OS X v10.3 and later.

Declared in `PMDefinitions.h`.

PostScript Injection Dictionary Keys

Constants that specify keys for PostScript injection dictionary entries.

```
#define kPSInjectionSectionKey     CFSTR("section")
#define kPSInjectionSubSectionKey  CFSTR("subsection")
#define kPSInjectionPageKey        CFSTR("page")
#define kPSInjectionPlacementKey   CFSTR("place")
#define kPSInjectionPostScriptKey  CFSTR("psdata")
```

Constants

`kPSInjectionSectionKey`

Specifies a section.

`kPSInjectionSubSectionKey`

Specifies a subsection.

`kPSInjectionPageKey`

Specifies a page.

`kPSInjectionPlacementKey`

Specifies placement.

`kPSInjectionPostScriptKey`

Specifies PostScript data.

PostScript Page Injection Options

Constants that specify PostScript injection options.

```
enum {
    kPSPageInjectAllPages = -1,
    kPSInjectionMaxDictSize = 5
};
```

Constants

`kPSPageInjectAllPages`

Specifies to inject all pages in the print job with PostScript code.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPSInjectionMaxDictSize`

Specifies the maximum size needed for a dictionary used for PostScript injection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

PostScript Injection Placement Options

Constants that specify where in the print job to inject PostScript code.

```
typedef UInt16 PSInjectionPlacement;
enum {
    kPSInjectionBeforeSubsection = 1,
    kPSInjectionAfterSubsection = 2,
    kPSInjectionReplaceSubsection = 3
};
```

Constants

`kPSInjectionBeforeSubsection`

Specifies that your PostScript code be inserted before the standard PostScript code that is normally emitted for the subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPSInjectionAfterSubsection`

Specifies that your PostScript code be inserted after the standard PostScript code that is normally emitted for the subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPSInjectionReplaceSubsection`

Specifies that your PostScript code replace the standard PostScript code that is normally emitted for the subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

PostScript Injection Sections

Constants that specify keys for PostScript injection section values.

```
typedef SInt32 PSInjectionSection;
enum {
    kInjectionSectJob = 1,
    kInjectionSectCoverPage = 2
};
```

Constants

`kInjectionSectJob`

Specifies the job section. This is the default section if you do not specify a section key explicitly.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSectCoverPage`

Specifies the cover page section. Currently unsupported.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

PostScript Injection Subsections

Constants that specify PostScript injection values for the subsection key.

```
typedef SInt32 PSInjectionSubsection;
enum {
    kInjectionSubPSAdobe = 1,
    kInjectionSubPSAdobeEPS = 2,
    kInjectionSubBoundingBox = 3,
    kInjectionSubEndComments = 4,
    kInjectionSubOrientation = 5,
    kInjectionSubPages = 6,
    kInjectionSubPageOrder = 7,
    kInjectionSubBeginProlog = 8,
    kInjectionSubEndProlog = 9,
    kInjectionSubBeginSetup = 10,
    kInjectionSubEndSetup = 11,
    kInjectionSubBeginDefaults = 12,
    kInjectionSubEndDefaults = 13,
    kInjectionSubDocFonts = 14,
    kInjectionSubDocNeededFonts = 15,
    kInjectionSubDocSuppliedFonts = 16,
    kInjectionSubDocNeededRes = 17,
    kInjectionSubDocSuppliedRes = 18,
    kInjectionSubDocCustomColors = 19,
    kInjectionSubDocProcessColors = 20,
    kInjectionSubPlateColor = 21,
    kInjectionSubPageTrailer = 22,
    kInjectionSubTrailer = 23,
    kInjectionSubEOF = 24,
    kInjectionSubBeginFont = 25,
    kInjectionSubEndFont = 26,
    kInjectionSubBeginResource = 27,
    kInjectionSubEndResource = 28,
    kInjectionSubPage = 29,
    kInjectionSubBeginPageSetup = 30,
    kInjectionSubEndPageSetup = 31
};
```

Constants

`kInjectionSubPSAdobe`

Specifies the “%!PS-Adobe” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPSAdobeEPS`

Specifies the “%!PS-Adobe-3.0 EPSF-3.0” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBoundingBox`

Specifies the “%BoundingBox” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndComments`

Specifies the "%EndComments" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubOrientation`

Specifies the "%Orientation" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPages`

Specifies the "%Pages" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPageOrder`

Specifies the "%PageOrder" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginProlog`

Specifies the "%BeginProlog" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndProlog`

Specifies the "%EndProlog" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginSetup`

Specifies the "%BeginSetup" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndSetup`

Specifies the "%EndSetup" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginDefaults`

Specifies the “%BeginDefaults” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndDefaults`

Specifies the “%EndDefaults” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocFonts`

Specifies the “%DocumentFonts” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocNeededFonts`

Specifies the “%DocumentNeededFonts” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocSuppliedFonts`

Specifies the “%DocumentSuppliedFonts” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocNeededRes`

Specifies the “%DocumentNeededResources” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocSuppliedRes`

Specifies the “%DocumentSuppliedResources” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocCustomColors`

Specifies the “%DocumentCustomColors” subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubDocProcessColors`

Specifies the "%DocumentProcessColors" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPlateColor`

Specifies the "%PlateColor" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPageTrailer`

Specifies the "%PageTrailer" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubTrailer`

Specifies the "%Trailer" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEOF`

Specifies the "%EOF" (end of file) subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginFont`

Specifies the "%BeginFont" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndFont`

Specifies the "%EndFont" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginResource`

Specifies the "%BeginResource" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndResource`

Specifies the "%EndResource" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubPage`

Specifies the "%Page" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubBeginPageSetup`

Specifies the "%BeginPageSetup" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kInjectionSubEndPageSetup`

Specifies the "%EndPageSetup" subsection.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

PostScript Printer Description File Domains

Constants that specify the domains for PostScript printer description (PPD) files.

```
typedef UInt16 PMPPDDomain;
enum {
    kAllPPDDomains = 1,
    kSystemPPDDomain = 2,
    kLocalPPDDomain = 3,
    kNetworkPPDDomain = 4,
    kUserPPDDomain = 5,
    kCUPSPPDDomain = 6
};
```

Constants

`kAllPPDDomains`

Specifies all available domains.

Available in Mac OS X v10.3 and later.

Declared in `PMDefinitions.h`.

`kSystemPPDDomain`

Specifies the system domain.

Available in Mac OS X v10.3 and later.

Declared in `PMDefinitions.h`.

`kLocalPPDDomain`
: Specifies the local domain.

  Available in Mac OS X v10.3 and later.

  Declared in `PMDefinitions.h`.

`kNetworkPPDDomain`
: Specifies the network domain.

  Available in Mac OS X v10.3 and later.

  Declared in `PMDefinitions.h`.

`kUserPPDDomain`
: Specifies the user domain.

  Available in Mac OS X v10.3 and later.

  Declared in `PMDefinitions.h`.

`kCUPSPPDDomain`
: Specifies the CUPS domain.

  Available in Mac OS X v10.3 and later.

  Declared in `PMDefinitions.h`.

Print All Pages Constant

A constant that specifies that all pages of a document should be printed.

```
enum {
    kPMPrintAllPages = -1
};
```

Constants

`kPMPrintAllPages`
: Specifies that all pages of a document should be printed.

  Available in Mac OS X v10.0 and later.

  Declared in `PMDefinitions.h`.

Print Quality Modes

Constants that specify standard options for print quality.

```
typedef UInt32 PMQualityMode;
enum {
    kPMQualityLowest = 0,
    kPMQualityInkSaver = 1,
    kPMQualityDraft = 4,
    kPMQualityNormal = 8,
    kPMQualityPhoto = 11,
    kPMQualityBest = 13,
    kPMQualityHighest = 15
};
```

Constants

`kPMQualityLowest`

Specifies to use the lowest print quality available to the printer.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityInkSaver`

Specifies to use a mode that saves ink, even if it slows printing.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityDraft`

Specifies to print at the highest speed, with the amount of ink used as a secondary consideration.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityNormal`

Specifies a general usage mode that balances quality and speed.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityPhoto`

Specifies to optimize the quality of photos on the page, with speed not a concern.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityBest`

Specifies to get the best print quality for all objects and photos on a page.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

`kPMQualityHighest`

Specifies to use the highest print quality available to the printer.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

Print Queue States

Constants that specify the current state of a print queue.

```
typedef UInt16 PMPrinterState;
enum {
    kPMPrinterIdle = 3,
    kPMPrinterProcessing = 4,
    kPMPrinterStopped = 5
};
```

Constants

`kPMPrinterIdle`

Specifies the idle state.

Available in Mac OS X v10.2 and later.

Declared in `PMDefinitions.h`.

`kPMPrinterProcessing`

Specifies the processing state.

Available in Mac OS X v10.2 and later.

Declared in `PMDefinitions.h`.

`kPMPrinterStopped`

Specifies the stopped state.

Available in Mac OS X v10.2 and later.

Declared in `PMDefinitions.h`.

Printer Description Types

Constants that specify printer description types.

```
#define kPMPPDDescriptionType CFSTR("PMPPDDescriptionType")
```

Constants

`kPMPPDDescriptionType`

Specifies a PostScript printer description (PPD).

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

Tag Constants

Constants that specify values, such as minimum and maximum values, that your application can pass to or obtain from printing functions.

```
typedef UInt32 PMTag;
enum {
    kPMCurrentValue = 'curr',
    kPMDefaultValue = 'dflt',
    kPMMinimumValue = 'minv',
    kPMMaximumValue = 'maxv',
    kPMSourceProfile = 'srcp',
    kPMMinRange = 'mnrg',
    kPMMaxRange = 'mxrg',
    kPMMinSquareResolution = 'mins',
    kPMMaxSquareResolution = 'maxs',
    kPMDefaultResolution = 'dftr'
};
```

Constants

`kPMCurrentValue`

Specifies the current setting or value.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMDefaultValue`

Specifies the default setting or value.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMinimumValue`

Specifies the minimum setting or value.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMaximumValue`

Specifies the maximum setting or value.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMSourceProfile`

Specifies a ColorSync source profile.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMinRange`

Specifies the minimum resolution supported by the printer.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMaxRange`
Specifies the maximum resolution supported by the printer.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMinSquareResolution`
Specifies the minimum resolution setting for which the horizontal and vertical resolutions are equal.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMMaxSquareResolution`
Specifies the maximum resolution setting for which the horizontal and vertical resolutions are equal.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMDefaultResolution`
Specifies the default resolution setting for the printer (typically 72 dots per inch).

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

User Cancellation Constant

A constant that specifies an error value that indicates the user canceled a printing operation.

```
enum {
    kPMCancel = 128
};
```

Constants

`kPMCancel`
Specifies that the user clicked the Cancel button in a Print or Page Setup dialog.

Available in Mac OS X v10.0 and later.

Declared in `PMDefinitions.h`.

Discussion

This constant is provided for compatibility with old applications and printer drivers that expect the `iPrAbort` error code to be returned when the user cancels a printing operation.

The default idle function checks for Command-period keyboard events during printing, and sets the error condition equal to `kPMCancel` if one occurs. Your application can check for this condition using the `PMSessionError` function, and should cancel the print job if `kPMCancel` is returned.

If you supply your own idle function (not needed in Mac OS 8, 9, or X), your function must check for Command-period keyboard events, and set the error condition using the function `PMSessionSetError`.

Color Modes

Constants that specify a color mode to use for printing.

```
typedef UInt16 PMColorMode;
enum {
    kPMBlackAndWhite = 1,
    kPMGray = 2,
    kPMColor = 3,
    kPMColorModeDuotone = 4,
    kPMColorModeSpecialColor = 5
};
```

Constants

`kPMBlackAndWhite`

Specifies black-and-white mode.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMGray`

Specifies grayscale mode.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMColor`

Specifies color mode.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMColorModeDuotone`

Specifies two-channel color mode.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

`kPMColorModeSpecialColor`

Specifies to allow special colors such as metallic and light cyan.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `PMDefinitionsDeprecated.h`.

Discussion

These constants are used by functions that are deprecated.

Result Codes

This table lists the result codes defined for Core Printing.

Result Code	Value	Description
kPMGeneralError	-30870	An unspecified error occurred. Available in Mac OS X v10.0 and later.
kPMOutOfScope	-30871	Your application called this function out of sequence with other printing functions. Available in Mac OS X v10.0 and later.
kPMNoDefaultPrinter	-30872	The user has not specified a default printer. Available in Mac OS X v10.0 and later.
kPMNotImplemented	-30873	The function is not implemented. Available in Mac OS X v10.0 and later.
kPMNoSuchEntry	-30874	There is no entry to match your application's request. Available in Mac OS X v10.0 and later.
kPMInvalidPrintSettings	-30875	Your application passed an invalid print settings object. Available in Mac OS X v10.0 and later.
kPMInvalidPageFormat	-30876	Your application passed an invalid page format object. Available in Mac OS X v10.0 and later.
kPMValueOutOfRange	-30877	Your application passed an out-of-range value. Available in Mac OS X v10.0 and later.
kPMLockIgnored	-30878	The lock value was ignored. Available in Mac OS X v10.0 and later.
kPMInvalidPrintSession	-30879	Your application passed an invalid printing session object. Available in Mac OS X v10.0 and later.
kPMInvalidPrinter	-30880	Your application passed an invalid printer object. Available in Mac OS X v10.0 and later.
kPMObjectInUse	-30881	The specified object is in use. Available in Mac OS X v10.0 and later.
kPMInvalidIndex	-30882	An array index is invalid. Available in Mac OS X v10.0 and later.
kPMStringConversionFailure	-30883	An internal error occurred while converting a string. Available in Mac OS X v10.0 and later.
kPMXMLParseError	-30884	An error occurred while parsing XML data. Available in Mac OS X v10.0 and later.

Result Code	Value	Description
kPMInvalidJobTemplate	-30885	An internal error occurred while creating a job template. Available in Mac OS X v10.0 and later.
kPMInvalidPrinterInfo	-30886	The printer information is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidConnection	-30887	The printer connection type is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidKey	-30888	The key in a ticket, job template, or dictionary is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidValue	-30889	The value in a ticket, job template, or dictionary is missing. Available in Mac OS X v10.0 and later.
kPMInvalidAllocator	-30890	The specified memory allocator is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidTicket	-30891	The job ticket is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidItem	-30892	The item being added to a ticket is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidType	-30893	The data type in a ticket, job template, or dictionary is not the expected type. Available in Mac OS X v10.0 and later.
kPMInvalidReply	-30894	A remote server or client sent an invalid reply. Available in Mac OS X v10.0 and later.
kPMInvalidFileType	-30895	The file type is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidObject	-30896	The object is invalid. Available in Mac OS X v10.0 and later.
kPMInvalidPaper	-30897	Your application passed an invalid paper object. Available in Mac OS X v10.2 and later.
kPMInvalidCalibrationTarget	-30898	The dictionary specifying a printer calibration target is invalid. Available in Mac OS X v10.3 and later.
kPMInvalidPreset	-30899	Your application passed an invalid preset object. Available in Mac OS X v10.3 and later.

Deprecated Core Printing Functions

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

Deprecated in Mac OS X v10.4

DisposePMIdleUPP

Disposes of a universal procedure pointer (UPP) to an idle callback. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
void DisposePMIdleUPP (
   PMIdleUPP userUPP
);
```

Discussion
You do not need this function in Mac OS X. Instead, use the standard idle proc. See the `PMIdleProcPtr` (page 106) callback function.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

InvokePMIdleUPP

Calls an idle callback. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
void InvokePMIdleUPP (
   PMIdleUPP userUPP
);
```

Discussion
You do not need this function in Mac OS X. Instead, use the standard idle proc. See the `PMIdleProcPtr` (page 106) callback function.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

NewPMIdleUPP

Creates a new universal procedure pointer (UPP) to an idle callback. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
PMIdleUPP NewPMIdleUPP (
   PMIdleProcPtr userRoutine
);
```

Discussion
You do not need this function in Mac OS X. Instead, use the standard idle proc. See the `PMIdleProcPtr` (page 106) callback function.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMBegin

Prepares Core Printing for use. (Deprecated in Mac OS X v10.4. Use `PMCreateSession` (page 31) instead.)

```
OSStatus PMBegin ();
```

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Do not nest calls to `PMBegin`.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMConvertOldPrintRecord

Creates a new `PMPageFormat` object and a new `PMPrintSettings` object from a print record created by the classic Printing Manager. (Deprecated in Mac OS X v10.4. There is no replacement; during the transition from Mac OS 9 to Mac OS X, this function facilitated the migration of print records saved in documents created in Mac OS 9, but the function no longer serves any useful purpose in Mac OS X.)

```
OSStatus PMConvertOldPrintRecord (
    Handle printRecordHandle,
    PMPrintSettings *printSettings,
    PMPageFormat *pageFormat
);
```

Parameters

printRecordHandle

A handle to a print record created by the classic Printing Manager.

printSettings

On return, a validated `PMPrintSettings` object.

pageFormat

On return, a validated `PMPageFormat` object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMDefaultPageFormat

Assigns default parameter values to an existing `PMPageFormat` object, for the current printer. (Deprecated in Mac OS X v10.4. Use `PMSessionDefaultPageFormat` (page 85) instead.)

```
OSStatus PMDefaultPageFormat (
    PMPageFormat pageFormat
);
```

Parameters

pageFormat

On return, a `PMPageFormat` object containing default parameter values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin` and creating a page format object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMDefaultPrintSettings

Assigns default parameter values to a `PMPrintSettings` object. (Deprecated in Mac OS X v10.4. Use `PMSessionDefaultPrintSettings` (page 85) instead.)

```
OSStatus PMDefaultPrintSettings (
    PMPrintSettings printSettings
);
```

Parameters

printSettings

A `PMPrintSettings` object. On return, the object contains default parameter values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin` and creating a print settings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMDisableColorSync

Disables ColorSync color matching for the current page. (Deprecated in Mac OS X v10.4. There is no replacement; draw using Quartz 2D instead.)

```
OSStatus PMDisableColorSync ();
```

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid between calls to `PMBeginPage` and `PMEndPage`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMDisposePageFormat

Releases memory previously allocated for a `PMPageFormat` object. (Deprecated in Mac OS X v10.4. Use `PMRelease` (page 76) instead.)

```
OSStatus PMDisposePageFormat (
    PMPageFormat pageFormat
);
```

Parameters

pageFormat

On return, an invalidated PMPageFormat object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin and creating a page format object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMDisposePrintSettings

Releases memory previously allocated for a PMPrintSettings object. (Deprecated in Mac OS X v10.4. Use PMRelease (page 76) instead.)

```
OSStatus PMDisposePrintSettings (
    PMPrintSettings printSettings
);
```

Parameters

printSettings

On return, an invalidated PMPrintSettings reference.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin and creating a print settings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMEnableColorSync

Enables ColorSync color matching for the current page. (Deprecated in Mac OS X v10.4. There is no replacement; draw using Quartz 2D instead.)

```
OSStatus PMEnableColorSync ();
```

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid between calls to PMBeginPage and PMEndPage.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMEnd

Closes Core Printing and releases its allocated memory. (Deprecated in Mac OS X v10.4. Use PMRelease (page 76) to release a PMPrintSession (page 110) object instead.)

```
OSStatus PMEnd (void);
```

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMError

Obtains the result code from the last printing function called by your application. (Deprecated in Mac OS X v10.4. Use PMSessionError (page 87) instead.)

```
OSStatus PMError ();
```

Return Value
A result code. The result code kPMCancel indicates the user canceled the current print job.

Discussion
Valid after calling PMBegin.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMGeneral

Maintains compatibility with the PrGeneral function in the classic Printing Manager. (Deprecated in Mac OS X v10.4. Use PMPrinterGetCommInfo (page 54) instead.)

```
OSStatus PMGeneral (
    Ptr pData
);
```

Parameters

pData

A pointer to a PrGeneral data structure.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMGetColorMode

Obtains the color mode for the print job. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
OSStatus PMGetColorMode (
   PMPrintSettings printSettings,
   PMColorMode *colorMode
);
```

Parameters

printSettings

The print settings object whose color mode you want to obtain.

colorMode

On return, a pointer to a value that represents the color mode setting. See "Color Modes" (page 131) for a list of possible return values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is not recommended. It doesn't do anything in Mac OS X and in general is no longer appropriate for applications to call.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMGetDriverCreator

Obtains the creator of the driver associated with the current printer. (Deprecated in Mac OS X v10.4. Use `PMPrinterGetDriverCreator` (page 55) instead.)

```
OSStatus PMGetDriverCreator (
    OSType *creator
);
```

Parameters

`creator`

On return, the 4-byte creator type of the driver (for example, `'APPL'` for an Apple printer driver).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMGetDriverReleaseInfo

Obtains release information for the driver associated with the current printer. (Deprecated in Mac OS X v10.4. Use `PMPrinterGetDriverReleaseInfo` (page 55) instead.)

```
OSStatus PMGetDriverReleaseInfo (
    VersRec *release
);
```

Parameters

`release`

On return, a pointer to a `VersRec` data structure containing the driver's short and long version strings and country code.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMGetGrafPtr

Obtains the printing port from an opaque printing context. (Deprecated in Mac OS X v10.4. Use `PMSessionGetCGGraphicsContext` (page 88) instead.)

```
OSStatus PMGetGrafPtr (
    PMPrintContext printContext,
    GrafPtr *grafPort
);
```

Parameters

printContext

A `PMPrintContext` object.

grafPort

On return, a pointer to a `grafPort` defining the current printing port.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid after calling `PMBegin` and creating a printing context.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMGetIndexedPrinterResolution

Obtains a resolution setting based on an index into the range of settings supported by the current printer. (Deprecated in Mac OS X v10.4. Use `PMPrinterGetIndexedPrinterResolution` (page 57) instead.)

```
OSStatus PMGetIndexedPrinterResolution (
    UInt32 index,
    PMResolution *res
);
```

Parameters

index

An index into the range of resolution settings supported by the specified printer. Index values begin at 1.

res

On return, the printer resolution setting.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin. You must first use the PMGetPrinterResolutionCount function to obtain the number of resolution settings supported by the current printer.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMGetJobName

Obtains the name of the print job. (Deprecated in Mac OS X v10.4. Use PMPrintSettingsGetJobName (page 72) instead.)

```
OSStatus PMGetJobName (
   PMPrintSettings printSettings,
   StringPtr name
);
```

Parameters

printSettings

A PMPrintSettings object.

name

On return, the name of the print job.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin and creating a print settings object. Before using this function you must call PMValidatePrintSettings to ensure that the print settings object is valid.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMGetLanguageInfo

Obtains information about the current printer's imaging language. (Deprecated in Mac OS X v10.4. Use PMPrinterGetLanguageInfo (page 57) instead.)

```
OSStatus PMGetLanguageInfo (
    PMLanguageInfo *info
);
```

Parameters

info

On return, a pointer to a data structure containing the printer's language level, version and release. The format of the returned data is based on the PostScript language.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin. PMGetLanguageInfo is useful for PostScript printers but may be irrelevant for other types of printers.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMGetPhysicalPageSize

Obtains the size of the imageable area in points, unaffected by rotation, resolution, or scaling. (Deprecated in Mac OS X v10.4. Use PMGetUnadjustedPageRect (page 40) or examine the paper returned by PMGetPageFormatPaper (page 38).)

```
OSStatus PMGetPhysicalPageSize (
   PMPageFormat pageFormat,
   PMRect *pageSize
);
```

Parameters

pageFormat

A PMPageFormat object previously created by your application.

pageSize

On return, a rectangle describing the physical page size where your application can draw.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMGetPhysicalPaperSize

Obtains the size of the paper in points, unaffected by rotation, resolution, or scaling. (Deprecated in Mac OS X v10.4. Use `PMGetUnadjustedPaperRect` (page 41) or examine the paper returned by `PMGetPageFormatPaper` (page 38).)

```
OSStatus PMGetPhysicalPaperSize (
   PMPageFormat pageFormat,
   PMRect *paperSize
);
```

Parameters

pageFormat

A `PMPageFormat` object previously created by your application.

paperSize

On return, a rectangle describing the physical size of the paper. Units are in 1/72 inch. Thus a 8.5 x 11 sheet of paper returns for its individual components:

top - 0.0

left - 0.0

bottom - 792.0

right - 612.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMGetPrinterResolution

Obtains the resolution setting for the current printer according to the tag parameter. (Deprecated in Mac OS X v10.4. Use `PMPrinterGetPrinterResolutionCount` (page 61) and `PMPrinterGetIndexedPrinterResolution` (page 57) to examine the available printer resolutions.)

```
OSStatus PMGetPrinterResolution (
    PMTag tag,
    PMResolution *res
);
```

Parameters

tag

Specifies the kind of resolution information required.

res

The printer resolution setting.

Return Value

A result code. . The result code `kPMNotImplemented` indicates that the printer driver does not support multiple resolution settings.

Discussion

Valid after calling PMBegin.

The following resolution tag constants are recognized:

kPMMinRange

The minimum resolution supported by the printer.

kPMMaxRange

The maximum resolution supported by the printer.

kPMMinSquareResolution

The minimum resolution setting for which the horizontal and vertical resolutions are equal.

kPMMaxSquareResolution

The maximum resolution setting for which the horizontal and vertical resolutions are equal.

kPMDefaultResolution

The default resolution setting for the printer (typically 72 dpi).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMGetPrinterResolutionCount

Obtains the number of resolution settings supported by the current printer. (Deprecated in Mac OS X v10.4. Use PMPrinterGetPrinterResolutionCount (page 61) instead.)

```
OSStatus PMGetPrinterResolutionCount (
    UInt32 *count
);
```

Parameters

count

On return, the number of supported printing resolutions.

Return Value

A result code. The result code kPMNotImplemented indicates that the printer driver does not support multiple resolution settings.

Discussion

Valid after calling PMBegin.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMIsPostScriptDriver

Reports whether the current printer driver supports the PostScript language. (Deprecated in Mac OS X v10.4. Use `PMPrinterIsPostScriptCapable` (page 64) or `PMPrinterIsPostScriptPrinter` (page 64) instead.)

```
OSStatus PMIsPostScriptDriver (
    Boolean *isPostScript
);
```

Parameters

isPostScript

Returns `true` if the current printer driver supports PostScript.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin`. In Mac OS X, `PMIsPostScriptDriver` always returns `false`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMMakeOldPrintRecord

Creates an old-style print record from a `PMPageFormat` and a `PMPrintSettings` object. (Deprecated in Mac OS X v10.4. There is no replacement; old-style print records are obsolete and serve no useful purpose in Mac OS X.)

```
OSStatus PMMakeOldPrintRecord (
    PMPrintSettings printSettings,
    PMPageFormat pageFormat,
    Handle *printRecordHandle
);
```

Parameters

printSettings

A `PMPrintSettings` object.

pageFormat

A `PMPageFormat` object.

printRecordHandle

On return, a handle to a an old-style print record. Your application must dispose of this handle.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling `PMBegin` and creating a page format and print settings object. Use `PMMakeOldPrintRecord` to create a print record to store with your documents for compatibility with pre-Carbon versions of your application. Note that because the page format and print settings objects contain more information than the old-style print record, some settings may be lost in conversion.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMNewPageFormat

Creates a new `PMPageFormat` object. (Deprecated in Mac OS X v10.4. Use `PMCreatePageFormat` (page 29) instead.)

```
OSStatus PMNewPageFormat (
    PMPageFormat *pageFormat
);
```

Parameters

pageFormat

On return, an initialized `PMPageFormat` object.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid after calling `PMBegin`. The function `PMNewPageFormat` allocates memory for a new `PMPageFormat` object in your application's memory space. The new page format object is empty until you set its values, or until you call `PMDefaultPageFormat` or `PMValidatePageFormat`.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMNewPrintSettings

Creates a new `PMPrintSettings` object. (Deprecated in Mac OS X v10.4. Use `PMCreatePrintSettings` (page 30) instead.)

```
OSStatus PMNewPrintSettings (
    PMPrintSettings *printSettings
);
```

Parameters

printSettings

On return, an initialized `PMPrintSettings` object.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid after calling PMBegin. The function PMNewPrintSettings allocates memory for a new PMPrintSettings object in your application's memory space. The new print settings object is empty until you set its values, or until you call PMDefaultPrintSettings or PMValidatePrintSettings.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMPostScriptBegin

Puts the current driver into PostScript mode, ready to accept PostScript data instead of QuickDraw data. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

```
OSStatus PMPostScriptBegin ();
```

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid between calls to PMBeginPage and PMEndPage. Call PMIsPostScriptDriver before calling PMPostScriptBegin to ensure that the current driver supports PostScript data.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMPostScriptData

Passes PostScript data, referenced by a pointer, to the current printer driver. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

```
OSStatus PMPostScriptData (
    Ptr psPtr,
    Size len
);
```

Parameters

psPtr

A pointer to PostScript data.

len

The number of bytes of PostScript data to pass to the current driver.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid between calls to PMPostScriptBegin and PMPostScriptEnd.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMPostScriptEnd

Restores the current driver to QuickDraw mode, ready to accept QuickDraw data instead of PostScript data. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

```
OSStatus PMPostScriptEnd ();
```

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid between calls to PMBeginPage and PMEndPage. Call PMPostScriptEnd to complete a PostScript session started with PMPostScriptBegin.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
PMCoreDeprecated.h

PMPostScriptFile

Passes PostScript data, contained in a file, to the current printer driver. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

```
OSStatus PMPostScriptFile (
    FSSpec *psFile
);
```

Parameters

psFile

A file specification.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid between calls to `PMPostScriptBegin` and `PMPostScriptEnd`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMPostScriptHandle

Passes PostScript data, referenced by a handle, to the current printer driver. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMPostScriptHandle (
    Handle psHandle
);
```

Parameters

psHandle

A reference to PostScript data.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid between calls to `PMPostScriptBegin` and `PMPostScriptEnd`.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMPrinterGetDescriptionURL

Obtains a reference to the specified printer's description file. (Deprecated in Mac OS X v10.4. Use `PMPrinterCopyDescriptionURL` (page 51) instead.)

```
OSStatus PMPrinterGetDescriptionURL (
   PMPrinter printer,
   CFStringRef descriptionType,
   CFURLRef *fileURL
);
```

Parameters

printer

The printer whose description file you want to obtain.

descriptionType

A Core Foundation string that specifies the type of description file for the selected printer. Currently, there is only one type defined—`kPMPPDDescriptionType`.

fileURL

A pointer to your `CFURLRef` variable. On return, the variable refers to a URL for the printer's description file. In spite of the name, the function `PMPrinterGetDescriptionURL` has Create/Copy semantics which means the caller must release the returned URL if it is not `NULL` and the result code `noErr` is returned.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can use this function to obtain a reference to the PostScript printer description (PPD) file for a PostScript printer.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMPrinterGetDeviceURI

Obtains a copy of a printer's device URI. (Deprecated in Mac OS X v10.4. Use `PMPrinterCopyDeviceURI` (page 52) instead.)

```
OSStatus PMPrinterGetDeviceURI (
   PMPrinter printer,
   CFURLRef *deviceURI
);
```

Parameters

printer

The printer whose device URI you want to obtain.

deviceURI

A pointer to your `CFURLRef` variable. On return, the variable refers to a URI for the location of the printer device. In spite of the name, this function has Create/Copy semantics which means the caller must release the returned URL if it is not `NULL` and the result code `noErr` is returned.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionConvertOldPrintRecord

Creates new page format and print settings objects from an old-style print record created for the classic Printing Manager. (Deprecated in Mac OS X v10.4. There is no replacement; during the transition from Mac OS 9 to Mac OS X, this function facilitated the migration of print records saved in documents created in Mac OS 9, but the function no longer serves any useful purpose in Mac OS X.)

```
OSStatus PMSessionConvertOldPrintRecord (
   PMPrintSession printSession,
   Handle printRecordHandle,
   PMPrintSettings *printSettings,
   PMPageFormat *pageFormat
);
```

Parameters

printSession

The current printing session.

printRecordHandle

A handle to an old-style print record created by the classic Printing Manager. You are responsible for disposing of the handle.

printSettings

On return, a print settings object that contains values converted from the print record. You are responsible for releasing the print settings object with the function PMRelease (page 76).

pageFormat

On return, a page format object that contains values converted from the print record. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31).

You can use PMSessionConvertOldPrintRecord to create page format and print settings objects from old-style print records stored in documents created by pre-Carbon versions of your application. You should validate the page format and print settings objects returned to you by calling the functions PMSessionValidatePageFormat and PMSessionValidatePrintSettings. Note that perfect translation between the old and new style objects is not achievable.

In Mac OS X, the function assumes the print record to be converted is a LaserWriter 8 print record.

Special Considerations

If you need to convert a Mac OS 9 print record into data you can use in Mac OS X, you should extract the page size data from the print record and use the function PMCreatePageFormatWithPMPaper (page 30) to create a PMPageFormat object that corresponds to that data.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionGeneral

Maintains compatibility with the PrGeneral function in the classic Printing Manager. (Deprecated in Mac OS X v10.4. Use PMPrinterGetCommInfo (page 54) instead.)

```
OSStatus PMSessionGeneral (
   PMPrintSession printSession,
   Ptr pData
);
```

Parameters

printSession

The printing session whose data you want to obtain.

pData

A pointer to a PrGeneral data structure.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The function PMSessionGeneral is valid for the printing session passed to the function. In Mac OS X, the function PMSessionGeneral makes an attempt to get the requested data if the opcode is getPSInfoOp. Otherwise the result code kPMNotImplemented is returned.

For more information about using the function PMSessionGeneral, see *Supporting Printing in Your Carbon Application*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionGetDocumentFormatGeneration

Obtains the spool file formats that can be generated for the specified printing session. (Deprecated in Mac OS X v10.4. If you're drawing using Quartz 2D instead of QuickDraw, use PMSessionBeginCGDocument or PMSessionBeginCGDocumentNoDialog (page 79); for submitting PostScript data, use PMPrinterPrintWithFile (page 65) or PMPrinterPrintWithProvider (page 66); to draw EPS data, use PMCGImageCreateWithEPSDataProvider (page 25).)

```
OSStatus PMSessionGetDocumentFormatGeneration (
   PMPrintSession printSession,
   CFArrayRef *docFormats
);
```

Parameters

printSession

The printing session whose spool file formats you want to obtain.

docFormats

A pointer to your CFArrayRef variable. On return, the variable refers to a Core Foundation array that contains the MIME types for the available spool file formats. Each element in the array is a Core Foundation string. Despite what its name implies, the function PMSessionGetDocumentFormatGeneration has Create/Copy semantics which means you are responsible for releasing the array.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call the function PMSessionGetDocumentFormatGeneration between the creation and release of a printing session. See the function PMCreateSession (page 31). You should call PMSessionGetDocumentFormatGeneration only after the Print dialog is dismissed.

The function PMSessionGetDocumentFormatGeneration determines the spool file formats that the specific print job supports. Spool file formats are represented by MIME types. The Mac OS X print spooler supports PDF and PICT + PS. The default spool file format is PDF. PICT + PS is supported only for printing to a PostScript printer.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

See Also

PMSessionSetDocumentFormatGeneration (page 163)

Declared In

PMCoreDeprecated.h

PMSessionGetDocumentFormatSupported

Obtains the spool file formats that are accepted by the current printer driver. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65), PMPrinterPrintWithProvider (page 66), or PMCGImageCreateWithEPSDataProvider (page 25) instead.)

```
OSStatus PMSessionGetDocumentFormatSupported (
   PMPrintSession printSession,
   CFArrayRef *docFormats,
   UInt32 limit
);
```

Parameters

printSession

The current printing session.

docFormats

On return, an array of CFString values containing MIME types specifying the spool file formats supported by the current printer driver. See "Document Format Strings" for a description of possible return values.

limit

The maximum number of supported document formats to be returned.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid within the context of a printing session.

Spool file formats are represented by MIME types. In Mac OS X, printer modules may support a wide range of spool file formats. The first item in the list of supported spool file formats is the default for the current printer driver.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionIsDocumentFormatSupported

Reports whether the current printer driver supports a specified spool file format. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionIsDocumentFormatSupported (
   PMPrintSession printSession,
   CFStringRef docFormat,
   Boolean *supported
);
```

Parameters

printSession

The current printing session.

docFormat

A spool file format represented by a MIME type.

supported

Returns `true` if the spool file format is supported by the current printer driver.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid within the context of a printing session.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionMakeOldPrintRecord

Creates an old-style print record from page format and print settings objects. (Deprecated in Mac OS X v10.4. There is no replacement; old-style print records are obsolete and serve no useful purpose in Mac OS X.)

```
OSStatus PMSessionMakeOldPrintRecord (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   PMPageFormat pageFormat,
   Handle *printRecordHandle
);
```

Parameters

printSession

The current printing session.

printSettings

A print settings object. To create a print settings object you can call the function PMCreatePrintSettings (page 30) and then call the function PMSessionDefaultPrintSettings (page 85) to initialize the print settings object to default values.

pageFormat

A page format object. To create a page format object you can call the function PMCreatePageFormat (page 29) and then call the function PMSessionDefaultPageFormat (page 85) to initialize the page format object to default values.

printRecordHandle

On return, a handle to an old-style print record. You are responsible for disposing of the handle.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31).

You can use PMSessionMakeOldPrintRecord to create an old-style print record to store with your documents for compatibility with pre-Carbon versions of your application. Note that because the page format and print settings objects contain more information than the old print record, some settings may be lost in the conversion. That is, perfect translation between the old and new style objects is not achievable.

In Mac OS X, the function always creates a LaserWriter 8 compatible print record.

Special Considerations

The proper way to keep page format information for use in Mac OS X is with a flattened PMPageFormat object. Typically applications don't keep print settings with a document but if that is appropriate for a given application, the proper way to do so is to use a flattened PMPrintSettings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionPostScriptBegin

Puts the current printer driver into PostScript mode, ready to accept PostScript data instead of QuickDraw data. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionPostScriptBegin (
   PMPrintSession printSession
);
```

Parameters

printSession

The current printing session.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call the function `PMSessionPostScriptBegin` between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You must also call the function within the scope of the functions `PMSessionBeginPage` and `PMSessionEndPage`.

To ensure that the current printer driver supports PostScript data, call `PMSessionGetDocumentFormatGeneration` before you call the function `PMSessionPostScriptBegin`. Check the list of supported spool file formats. If PICT + PS is one of them, select that format by calling the function `PMSessionSetDocumentFormatGeneration`. The function `PMSessionSetDocumentFormatGeneration` must be called before you call `PMSessionBeginDocument`.

The function `PMSessionPostScriptBegin` is not useful unless the current port is the printing port. The function returns `true` if the document format is not PICT + PS.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionPostScriptData

Passes PostScript data, referenced by a pointer, to the current printer driver. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionPostScriptData (
   PMPrintSession printSession,
   Ptr psPtr,
   Size len
);
```

Parameters

printSession

The current printing session.

psPtr

A pointer to the PostScript data you want to pass to the current printer driver.

len

The number of bytes of PostScript data.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). Typically you call this function with the scope of calls to the functions `PMSessionPostScriptBegin` and `PMSessionPostScriptEnd`.

The function `PMSessionPostScriptData` is not useful unless the current port is the printing port and the document format is PICT + PS.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionPostScriptEnd

Restores the current driver to QuickDraw mode, ready to accept QuickDraw data instead of PostScript data. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionPostScriptEnd (
   PMPrintSession printSession
);
```

Parameters

printSession

The current printing session.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion
You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You must also call this function with the scope of calls to the functions `PMSessionBeginPage` and `PMSessionEndPage`.

You call the function `PMSessionPostScriptEnd` to complete a PostScript block started with `PMSessionPostScriptBegin`. The function `PMSessionPostScriptEnd` is not useful unless the current port is the printing port and the document format is PICT + PS.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`PMCoreDeprecated.h`

PMSessionPostScriptFile

Passes the PostScript data, contained in a file, to the current printer driver. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionPostScriptFile (
   PMPrintSession printSession,
   FSSpec *psFile
);
```

Parameters

printSession

The current printing session.

psFile

A pointer to a variable that specifies a file location. The file should contain the PostScript data you want to pass to the current printer driver.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You typically call this function within the scope of calls to the function `PMSessionPostScriptBegin` and `PMSessionPostScriptEnd`.

The function `PMSessionPostScriptFile` is not useful unless the current port is the printing port and the document format is PICT + PS.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionPostScriptHandle

Passes the PostScript data, referenced by a Memory Manager handle, to the current printer driver. (Deprecated in Mac OS X v10.4. Use `PMPrinterPrintWithFile` (page 65), `PMPrinterPrintWithProvider` (page 66), or `PMCGImageCreateWithEPSDataProvider` (page 25) instead.)

```
OSStatus PMSessionPostScriptHandle (
   PMPrintSession printSession,
   Handle psHandle
);
```

Parameters

printSession

The current printing session.

psHandle

A handle to the PostScript data you want to pass to the current printer driver. You must make sure the handle is of the appropriate size for the data, otherwise you risk corrupting the spool file.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You must also call this function within the scope of calls to the function `PMSessionPostScriptBegin` and `PMSessionPostScriptEnd`.

The function `PMSessionPostScriptEnd` is not useful unless the current port is the printing port and the document format is PICT + PS.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionSetCurrentPrinter

Changes the current printer for a printing session to a printer specified by name. (Deprecated in Mac OS X v10.4. Use `PMSessionSetCurrentPMPrinter` (page 91) instead.)

```
OSStatus PMSessionSetCurrentPrinter (
   PMPrintSession session,
   CFStringRef printerName
);
```

Parameters

session

The printing session whose printer you want to change.

printerName

The name of the printer you want to set as the current printer.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31).

Availability

Available in Mac OS X v10.1 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionSetDocumentFormatGeneration

Requests a specified spool file format and supplies the graphics context type to use for drawing pages within the print loop. (Deprecated in Mac OS X v10.4. If you're drawing using Quartz 2D instead of QuickDraw, use `PMSessionBeginCGDocument` or `PMSessionBeginCGDocumentNoDialog` (page 79); for submitting PostScript data, use `PMPrinterPrintWithFile` (page 65) or `PMPrinterPrintWithProvider` (page 66); to draw EPS data, use `PMCGImageCreateWithEPSDataProvider` (page 25).)

```
OSStatus PMSessionSetDocumentFormatGeneration (
   PMPrintSession printSession,
   CFStringRef docFormat,
   CFArrayRef graphicsContextTypes,
   CFTypeRef options
);
```

Parameters

printSession

The printing session whose spool file format and graphics context type you want to specify.

docFormat

A Core Foundation string that specifies the desired spool file format as a MIME type. See "Document Format Strings" (page 115) for a description of the constants you can use to specify the document format.

graphicsContexts

A reference to a Core Foundation array of graphics contexts to use for drawing pages within the print loop. You can supply a QuickDraw graphics context (`kPMGraphicsContextQuickDraw`) or a Quartz 2D graphics context (`kPMGraphicsContextCoreGraphics`). An array of length 1 is the only length that is supported, regardless of graphics context type. See "Graphics Context Types" (page 116) for a description of the constants you can use to specify a graphics context.

options

Reserved for future use.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You only need to call the function `PMSessionSetDocumentFormatGeneration` if you want to specify a format other than the default format (PDF) or a graphics context other than the default context (QuickDraw). If you want to use the default format for the operating system and to draw with QuickDraw, then you do not need to call this function. If you want to generate PICT + PS to use as one of the supported formats, then call `PMSessionSetDocumentFormatGeneration` to set the graphics context to QuickDraw and the format to PICT + PS. Note that the PICT + PS format is not available on Intel-based systems.

If you want to use a Quartz 2D graphics context to draw each page, you can call the following code to inform the printing system in all versions of Mac OS X.

```
static OSStatus MyPMSessionBeginCGDocument (
    PMPrintSession printSession,
    PMPrintSettings printSettings,
    PMPageFormat pageFormat)
{
    OSStatus err = noErr;

    // Use the simpler call if it is present.
    if(&PMSessionBeginCGDocument != NULL) {
       err = PMSessionBeginCGDocument (printSession, printSettings, pageFormat);
    }
    else {
        CFStringRef s[1] = { kPMGraphicsContextCoreGraphics };
        CFArrayRef graphicsContextsArray = CFArrayCreate (
            kCFAllocatorDefault, (const void**)s, 1, &kCFTypeArrayCallBacks);
        err = PMSessionSetDocumentFormatGeneration (
            printSession, kPMDocumentFormatPDF, graphicsContextsArray, NULL);
        CFRelease (graphicsContextsArray);
        if(!err)
            err = PMSessionBeginDocument (
                printSession, printSettings, pageFormat);
    }
    return err;
}
```

The previous code informs the printing system that you want a Quartz graphics context, but you get the actual context for your printing port only after you call the function `PMSessionBeginPage` and then call the following code.

```
static OSStatus MyPMSessionGetCGGraphicsContext (
    PMPrintSession printSession,
    CGContextRef *printingContextP)
{
    OSStatus err = noErr;
    // Use the simpler call if it is present.
    if(&PMSessionGetCGGraphicsContext != NULL) {
        err = PMSessionGetCGGraphicsContext (printSession, printingContextP);
    }
    else {
        err = PMSessionGetGraphicsContext (
            printSession, kPMGraphicsContextCoreGraphics,
            (void**)printingContextP);
    }
    return err;
}
```

The printing context you get is a Quartz context into which you can draw. Note that the default coordinate system for Quartz 2D is not the same as that used for QuickDraw. Quartz 2D defines the coordinates of the lower-left corner of the sheet as (0,0) whereas the origin for the QuickDraw port is the upper-left corner of the imageable area.

You must call this function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You must call the function `PMSessionSetDocumentFormatGeneration` before you call `PMSessionBeginDocument` or `PMSessionBeginDocumentNoDialog` (page 179). Before requesting a spool file format using this function, you should call the function `PMSessionGetDocumentFormatGeneration` to get the list of supported formats.

Special Considerations

The PICT + PS spool file format is not available on Intel-based systems.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

See Also

`PMSessionGetDocumentFormatGeneration` (page 155)

Related Sample Code

CarbonSketch

Declared In

`PMCoreDeprecated.h`

PMSessionSetIdleProc

Installs an idle callback function in your print loop. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
OSStatus PMSessionSetIdleProc (
   PMPrintSession printSession,
   PMIdleUPP idleProc
);
```

Parameters

printSession

The printing session that provides a context for the print job.

idleProc

A universal procedure pointer to your idle function. Your idle function is defined by the callback PMIdleProcPtr (page 106).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You do not need this function in Mac OS X. Instead, use the standard idle proc.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionSetPSInjectionData

Specifies a set of PostScript code injection points and the PostScript data to be injected. (Deprecated in Mac OS X v10.4. Use PMPrinterPrintWithFile (page 65) or PMPrinterPrintWithProvider (page 66) instead.)

```
OSStatus PMSessionSetPSInjectionData (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   CFArrayRef injectionDictArray
);
```

Parameters

printSession

The current printing session.

printSettings

The print settings object in which to place the specified injection points.

injectionDictArray

A reference to a Core Foundation array that contains one or more Core Foundation dictionary (CFDictionary) entries. Each dictionary entry specifies PostScript injection data you want inserted at a specific point in the print stream. See "PostScript Injection Dictionary Keys" (page 118) for a description of the constants you can use as keys for these dictionary entries.

Return Value

A result code. See "Core Printing Result Codes" (page 131). The result code kPMInvalidParameter is returned if the injectionDictArray object contains any invalid entries. The result code kPMInvalidPrintSession is returned if the document format has not been set to kPMDocumentFormatPICTPS for the specified printing session.

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). Before calling PMSessionSetPSInjectionData, your application must set the document format of the printing session to kPMDocumentFormatPICTPS using the function PMSessionSetDocumentFormatGeneration (page 163).

For applications that require extensive control over PostScript code generation, the function PMSessionSetPSInjectionData provides the ability to insert PostScript code into specified places in the print stream. It is intended for use by desktop publishing applications for which functions such as PMSessionPostScriptData do not provide sufficient control.

You specify the injection points by creating an array of CFDictionary entries. Each dictionary consists of key-value pairs in which the key specifies where to inject the PostScript and the value specifies the PostScript data you want to inject. The function verifies that the injectionDictArray parameter is properly formed, and returns the kPMInvalidParameter result code if the array does not contain valid entries.

If you call the function PMSessionSetPSInjectionData a second time for a given print settings object, the injection points you specified previously are replaced with the new injection points.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSetColorMode

Sets the desired color mode for the print job. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
OSStatus PMSetColorMode (
   PMPrintSettings printSettings,
   PMColorMode colorMode
);
```

Parameters

printSettings

The print settings object whose color mode you want to set.

colorMode

The desired color mode. See "Color Modes" (page 131) for a list of possible values.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Special Considerations

This function does nothing in Mac OS X.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`PMCoreDeprecated.h`

PMSetError

Sets the value of the current result code. (Deprecated in Mac OS X v10.4. Use `PMSessionSetError` (page 93) instead.)

```
OSStatus PMSetError (
    OSStatus printError
);
```

Parameters

`printError`

The result code you wish to set. This result code will be returned by the `PMError` function.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid after calling `PMBegin`.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In
`PMCoreDeprecated.h`

PMSetIdleProc

Installs an idle callback function in your print loop. (Deprecated in Mac OS X v10.4. There is no replacement; this function was included to facilitate porting legacy applications to Mac OS X, but it serves no useful purpose.)

```
OSStatus PMSetIdleProc (
    PMIdleUPP idleProc
);
```

Parameters

`idleProc`

A universal procedure pointer to your idle function. Your idle function is defined by the callback `PMIdleProcPtr`.

Return Value
A result code. See "Core Printing Result Codes" (page 131).

Discussion
Valid after calling `PMBegin`. The printing system calls your idle function periodically during your print loop.

Special Considerations

Your idle function is not called in Mac OS X. It's only called in Mac OS 8 and 9.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PMCoreDeprecated.h`

PMSetJobName

Specifies the name of the print job. (Deprecated in Mac OS X v10.4. Use `PMPrintSettingsSetJobName` (page 74) instead.)

```
OSStatus PMSetJobName (
   PMPrintSettings printSettings,
   StringPtr name
);
```

Parameters

`printSettings`

A `PMPrintSettings` object.

`name`

The name to assign to the print job. This string will be used to name the spool file.

Return Value

A result code. The result code `kPMInvalidParameter` is returned if you attempt to set the job name to an invalid file name or a null string.

Discussion

Valid after calling `PMBegin` and creating a print settings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSetPhysicalPaperSize

Requests a particular paper size, unaffected by rotation, resolution, or scaling. (Deprecated in Mac OS X v10.4. Use `PMCreatePageFormatWithPMPaper` (page 30) instead.)

```
OSStatus PMSetPhysicalPaperSize (
   PMPageFormat pageFormat,
   const PMRect *paperSize
);
```

Parameters

pageFormat

The PMPageFormat which will hold the new physical paper size.

paperSize

The desired paper size expressed as a PMRect. The units are 1/72 inch. A PMRect is a rectangle whose individual components are doubles.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMValidatePageFormat

Obtains a valid PMPageFormat object. (Deprecated in Mac OS X v10.4. Use PMSessionValidatePageFormat (page 93) instead.)

```
OSStatus PMValidatePageFormat (
    PMPageFormat pageFormat,
    Boolean *result
);
```

Parameters

pageFormat

A PMPageFormat object to be validated.

result

Returns true if any parameters were changed, false if no changes were required.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin and creating a page format object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

PMValidatePrintSettings

Obtains a valid PMPrintSettings object. (Deprecated in Mac OS X v10.4. Use PMSessionValidatePrintSettings (page 94) instead.)

```
OSStatus PMValidatePrintSettings (
    PMPrintSettings printSettings,
    Boolean *result
);
```

Parameters

printSettings

The PMPrintSettings object to be validated.

result

On return, a value of true if any parameters were changed, or false if no changes were required.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

Valid after calling PMBegin and creating a print settings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

PMCoreDeprecated.h

Deprecated in Mac OS X v10.5

PMFlattenPageFormat

Flattens a page format object into a Memory Manager handle for storage in a user document. (Deprecated in Mac OS X v10.5. Use PMPageFormatCreateDataRepresentation (page 41) instead.)

```
OSStatus PMFlattenPageFormat (
   PMPageFormat pageFormat,
   Handle *flatFormat
);
```

Parameters

pageFormat

The page format object to flatten.

flatFormat

A pointer to your Handle variable. On return, the variable refers to a Memory Manager handle that contains the flattened page format object. The handle is allocated by the function. You are responsible for disposing of the handle.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Related Sample Code

CarbonSketch

Declared In

`PMCoreDeprecated.h`

PMFlattenPageFormatToCFData

Flattens a page format object into a Core Foundation data object for storage in a user document. (Deprecated in Mac OS X v10.5. Use `PMPageFormatCreateDataRepresentation` (page 41) instead.)

```
OSStatus PMFlattenPageFormatToCFData (
   PMPageFormat pageFormat,
   CFDataRef *flatFormat
);
```

Parameters

`pageFormat`

The page format object to flatten.

`flatFormat`

A pointer to your `CFDataRef` variable. On return, the variable refers to a Core Foundation data object containing a flattened representation of the specified page format object. You are responsible for releasing the data object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

`PMUnflattenPageFormatWithCFData` (page 187)

Declared In

`PMCoreDeprecated.h`

PMFlattenPageFormatToURL

Flattens a page format object into a file for storage in a user document. (Deprecated in Mac OS X v10.5. Use `PMPageFormatCreateDataRepresentation` (page 41) and write the resulting data to your destination.)

```
OSStatus PMFlattenPageFormatToURL (
   PMPageFormat pageFormat,
   CFURLRef flattenFileURL
);
```

Parameters

pageFormat

The page format object to flatten.

flatFormat

A Core Foundation URL specifying a file to contain a flattened representation of the specified page format object. If the file already exists, it is overwritten. Only file-based URLs are supported.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

`PMUnflattenPageFormatWithURL` (page 188)

Declared In

`PMCoreDeprecated.h`

PMFlattenPrintSettings

Flattens a print settings object into a Memory Manager handle for storage in a user document. (Deprecated in Mac OS X v10.5. Use `PMPrintSettingsCreateDataRepresentation` (page 71) instead.)

```
OSStatus PMFlattenPrintSettings (
   PMPrintSettings printSettings,
   Handle *flatSettings
);
```

Parameters

printSettings

The print settings object to flatten.

flatSettings

A pointer to your `Handle` variable. On return, the variable refers to a Memory Manager handle that contains a flattened print settings object. The handle is allocated by the function. You are responsible for disposing of the handle.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

There are no scoping requirements as to when you may use this function.

Apple recommends that you do not reuse the print settings information if the user prints the document again. The information supplied by the user in the Print dialog should pertain to the document only while the document prints, so there is no need to save the print settings object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMFlattenPrintSettingsToCFData

Flattens a print settings object into a Core Foundation data object for storage in a user document. (Deprecated in Mac OS X v10.5. Use `PMPrintSettingsCreateDataRepresentation` (page 71) instead.)

```
OSStatus PMFlattenPrintSettingsToCFData (
   PMPrintSettings printSettings,
   CFDataRef *flatSetting
);
```

Parameters

`printSettings`

The print settings object to flatten.

`flatSetting`

A pointer to your `CFDataRef` variable. On return, the variable refers to a Core Foundation data object that contains a flattened representation of the specified print settings object. You are responsible for releasing the data object.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

`PMUnflattenPrintSettingsWithCFData` (page 189)

Declared In

`PMCoreDeprecated.h`

PMFlattenPrintSettingsToURL

Flattens a print settings object into a URL for storage in a user document. (Deprecated in Mac OS X v10.5. Instead use `PMPrintSettingsCreateDataRepresentation` (page 71) and write the resulting data to your destination.)

```
OSStatus PMFlattenPrintSettingsToURL (
   PMPrintSettings printSettings,
   CFURLRef flattenFileURL
);
```

Parameters

printSettings

The print settings object to flatten.

flattenFileURL

A Core Foundation URL specifying a file to contain a flattened representation of the specified print settings object. If the file already exists, it is overwritten. Only file-based URLs are supported.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

`PMUnflattenPrintSettingsWithURL` (page 190)

Declared In

`PMCoreDeprecated.h`

PMGetDestination

Obtains the output destination of a print job. (Deprecated in Mac OS X v10.5. Use `PMSessionGetDestinationType` (page 90), `PMSessionCopyDestinationFormat` (page 81), or `PMSessionCopyDestinationLocation` (page 82) instead.)

```
OSStatus PMGetDestination (
   PMPrintSettings printSettings,
   PMDestinationType *destType,
   CFURLRef *fileURL
);
```

Parameters

printSettings

The print settings object whose destination you want to obtain.

destType

A pointer to your `PMDestinationType` variable. On return, the variable indicates the destination for the print job. See "Destination Types" (page 114).

fileURL

A pointer to your `CFURLRef` variable. On return, the variable refers to a Core Foundation URL that contains the location of the print job destination.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

For print jobs that are sent to disk, as opposed a printer, you can use this function to obtain the location of the destination file. Valid within a printing session after creating a print settings object.

Before using this function you must call `PMSessionValidatePrintSettings` or `PMValidatePrintSettings` to ensure that the print settings object is valid.

Special Considerations

This function does not take a print session parameter and therefore cannot indicate whether preview has been selected as the destination.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMGetJobNameCFString

Obtains the name of the print job. (Deprecated in Mac OS X v10.5. Use `PMPrintSettingsGetJobName` (page 72) instead.)

```
OSStatus PMGetJobNameCFString (
   PMPrintSettings printSettings,
   CFStringRef *name
);
```

Parameters

`printSettings`

The print settings object whose job name you want to obtain.

`name`

A pointer to your `CFStringRef` variable. On return, the variable refers to a string that contains the name of the print job. Despite what its name implies, the function `PMGetJobNameCFString` has Create/Copy semantics which means your application must release the string returned to it.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

See Also

`PMSetJobNameCFString` (page 183)

Declared In

`PMCoreDeprecated.h`

PMGetResolution

Obtains the current application's drawing resolution. (Deprecated in Mac OS X v10.5. Draw using Quartz 2D and call `CGContextScaleCTM` instead.)

```
OSStatus PMGetResolution (
   PMPageFormat pageFormat,
   PMResolution *res
);
```

Parameters

pageFormat

The page format object whose drawing resolution you want to obtain.

res

A pointer to your PMResolution (page 111) structure. On return, the structure contains the drawing resolution of the current application.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function obtains the drawing resolution specified in the page format, not the resolution of the current printer. You can use PMPrinterGetPrinterResolutionCount (page 61) and PMPrinterGetIndexedPrinterResolution (page 57) to examine the available printer resolutions.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMPaperCreate

Creates a paper object. (Deprecated in Mac OS X v10.5. Use PMPrinterGetPaperList (page 61) to find the built-in papers available for a given printer or use PMPaperCreateCustom (page 43) to create a custom paper.)

```
OSStatus PMPaperCreate (
    PMPrinter printer,
    CFStringRef id,
    CFStringRef name,
    double width,
    double height,
    const PMPaperMargins *margins,
    PMPaper *paperP
);
```

Parameters

printer

A printer object for which the paper is appropriate.

id

A unique identifier for this paper.

name

The name to display to the user for this paper.

width

The width of the paper, in points.

height

The height of the paper, in points.

margins

A pointer to a PMPaperMargins (page 109) structure that specifies the unprintable margins of the paper, in points. The four values in the structure specify the top, left, bottom, and right imageable area margins of the paper.

paperP

A pointer to your PMPaper (page 109) variable. On return, the variable refers to a new paper object with the specified attributes. You are responsible for releasing the paper object with the function PMRelease (page 76). The variable is set to NULL if the object could not be created.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function creates a paper object appropriate for the specified printer. To obtain one of the available built-in paper sizes for a given printer, you should use the function PMPrinterGetPaperList (page 61).

Special Considerations

This function creates a paper object but does not mark it as a custom paper, so it appears to the printing system as if it were a built-in paper. This can be produce unpredictable results, so this function is deprecated.

Availability

Available in Mac OS X v10.3 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMPrinterGetPrinterResolution

Obtains a resolution setting for the specified printer. (Deprecated in Mac OS X v10.5. Use PMPrinterGetPrinterResolutionCount (page 61) and PMPrinterGetIndexedPrinterResolution (page 57) to examine the available printer resolutions.)

```
OSStatus PMPrinterGetPrinterResolution (
   PMPrinter printer,
   PMTag tag,
   PMResolution *res
);
```

Parameters

printer

The printer whose resolution you want to obtain.

tag

A tag that specifies the kind of resolution information you want to obtain (minimum, maximum, default, and so forth). See "Tag Constants" (page 128) for a description of the constants you can pass in this parameter.

res

A pointer to your PMResolution (page 111) data structure. On return, the structure contains the resolution setting associated with the tag value.

Return Value

A result code. The result code kPMNotImplemented indicates that the printer driver does not support multiple resolution settings.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionBeginDocumentNoDialog

Begins a print job that, by default, draws into a QuickDraw graphics port, and suppresses the printing status dialog. (Deprecated in Mac OS X v10.5. Use PMSessionBeginCGDocumentNoDialog (page 79) instead.)

```
OSStatus PMSessionBeginDocumentNoDialog (
   PMPrintSession printSession,
   PMPrintSettings printSettings,
   PMPageFormat pageFormat
);
```

Parameters

printSession

The printing session that provides a context for the new print job.

printSettings

The print settings to use for the new print job.

pageFormat

The page format to use for the new print job.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

The function PMSessionBeginDocumentNoDialog starts a print job and should be called within your application's print loop. This function is similar to the function PMSessionBeginDocument except that the printing status dialog is suppressed.

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). If you present a printing dialog before you call PMSessionBeginDocumentNoDialog, when calling this function you should use the same PMPrintSession (page 110) object you used to present the dialog.

Before you call PMSessionBeginDocumentNoDialog, you should call PMSessionValidatePrintSettings (page 94) and PMSessionValidatePageFormat (page 93) to make sure the specified print settings and page format objects are updated and valid. After you call PMSessionBeginDocumentNoDialog, if you call a function that changes the specified print settings or page format object, the change is ignored for the current print job.

This function must be called before its corresponding End function (PMSessionEndDocumentNoDialog (page 86)). If the function PMSessionBeginDocumentNoDialog returns noErr, you must call the End function, even if errors occur within the scope of the Begin and End functions.

The printing system automatically handles printing multiple copies. Your application does not need to perform any tasks other than specifying the number of copies in the printing session.

Special Considerations

In Mac OS X v10.4 and later, Apple recommends using the function `PMSessionBeginCGDocumentNoDialog` (page 79) instead of this function. QuickDraw is deprecated and your application should be using Quartz 2D for its rendering.

Availability

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionDisableColorSync

Disables use of a custom ColorSync profile previously enabled by the function `PMSessionEnableColorSync` (page 181). (Deprecated in Mac OS X v10.5. There is no replacement; draw using Quartz 2D instead.)

```
OSStatus PMSessionDisableColorSync (
   PMPrintSession printSession
);
```

Parameters

`printSession`

The printing session whose page-specific ColorSync profile you want to disable.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call the `PMSessionDisableColorSync` function between the creation and release of a printing session. See the function `PMCreateSession` (page 31). You must call this function within the scope of calls to the functions `PMSessionBeginPage` and `PMSessionEndPage`.

The function `PMSessionDisableColorSync` applies only to the current page. The function is useful only if the graphics context is QuickDraw and the current port is the printing port.

Special Considerations

This function is deprecated because QuickDraw is deprecated. When drawing with Quartz, the current stroke and fill color space and the color space associated with an image are used to characterize color. Quartz provides ways to use ColorSync profiles to create color spaces, so you can characterize color using ColorSync simply by drawing with Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSessionEnableColorSync

Enables use of a custom ColorSync profile previously set by the function PMSetProfile (page 184). (Deprecated in Mac OS X v10.5. There is no replacement; draw using Quartz 2D instead.)

```
OSStatus PMSessionEnableColorSync (
   PMPrintSession printSession
);
```

Parameters

printSession

The printing session whose page-specific ColorSync profile you want to enable.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). You must call this function within the scope of calls to the functions PMSessionBeginPage and PMSessionEndPage.

The function PMSessionEnableColorSync applies only to the current page. The function is useful only if the graphics context is QuickDraw and the current port is the printing port.

Special Considerations

This function is deprecated because QuickDraw is deprecated. When drawing with Quartz, the current stroke and fill color space and the color space associated with an image are used to characterize color. Quartz provides ways to use ColorSync profiles to create color spaces, so you can characterize color using ColorSync simply by drawing with Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSessionGetGraphicsContext

Obtains the graphics context for the current page in a printing session. (Deprecated in Mac OS X v10.5. Use PMSessionGetCGGraphicsContext (page 88) instead.)

```
OSStatus PMSessionGetGraphicsContext (
   PMPrintSession printSession,
   CFStringRef graphicsContextType,
   void **graphicsContext
);
```

Parameters

printSession

The printing session whose current graphics context you want to obtain.

graphicsType

The desired graphics context type. This parameter is currently ignored.

graphicsContext

On return, a reference to the current graphics context. The graphics context returned is the one last set by a call to the function PMSessionSetDocumentFormatGeneration or the default (QuickDraw) if there was no call to the function. You must typecast the context to an appropriate graphics type, either grafPtr or CGContextRef.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You must call this function between the creation and release of a printing session. See the function PMCreateSession (page 31). You must also call the function PMSessionGetGraphicsContext within the scope of the functions PMSessionBeginPage and PMSessionEndPage.

In Mac OS X v10.3 and earlier, you should call this function for each page you draw for a print job. After each call to the function PMSessionBeginPage your application should call PMSessionGetGraphicsContext to obtain the current graphics context. If that context is a QuickDraw context, then set the drawing port to this port by calling the QuickDraw SetPort function. See the discussion of the function PMSessionBeginPage for more information.

Special Considerations

In Mac OS X v10.4 and later, Apple recommends using the function PMSessionGetCGGraphicsContext (page 88) instead of this function. QuickDraw is deprecated and your application should be using Quartz 2D for its rendering.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Related Sample Code

CarbonSketch

Declared In

PMCoreDeprecated.h

PMSetAdjustedPageRect

Requests a particular page size, adjusted for the current rotation, resolution, or scaling settings. (Deprecated in Mac OS X v10.5. To set a particular paper size and margins, obtain or create a PMPaper (page 109) object and call PMCreatePageFormatWithPMPaper (page 30).)

```
OSStatus PMSetAdjustedPageRect (
   PMPageFormat pageFormat,
   const PMRect *pageRect
);
```

Parameters

pageFormat

The page format object whose page rectangle you want to set.

pageRect

A pointer to your PMRect (page 111) data structure that specifies the desired size of the page rectangle, in points. The top-left coordinates should be (0,0). See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

This function is not recommended. You should call this function only if your application provides desktop publishing and the Page Setup dialog does not provide sufficient control. Typically, such applications display their own specialized document format dialog.

If you decide to use this function, you must call the function between the creation and release of a printing session. See the function PMCreateSession (page 31). You can use PMSetAdjustedPageRect to set a drawing rectangle without going through the Page Setup dialog or calling other page format accessor functions. This function allows an application to specify the dimensions of the imageable area into which it draws.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMSetJobNameCFString

Specifies the name of a print job. (Deprecated in Mac OS X v10.5. Use PMPrintSettingsSetJobName (page 74) instead.)

```
OSStatus PMSetJobNameCFString (
   PMPrintSettings printSettings,
   CFStringRef name
);
```

Parameters

printSettings

The print settings object whose job name you want to set.

name

The new name for the print job.

Return Value

A result code. See "Core Printing Result Codes" (page 131). The result code kPMInvalidParameter is returned if you pass NULL or an empty string in the name parameter.

Discussion

You should call this function before you open the Print dialog.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

See Also

`PMGetJobNameCFString` (page 176)

Related Sample Code

CarbonSketch

Declared In

`PMCoreDeprecated.h`

PMSetProfile

Embeds a color profile during printing. (Deprecated in Mac OS X v10.5. There is no replacement; draw using Quartz 2D instead.)

```
OSStatus PMSetProfile (
   PMPrintSettings printSettings,
   PMTag tag,
   const CMProfileLocation *profile
);
```

Parameters

printSettings

The print settings object in which to embed the color profile.

tag

A tag that describes the usage of the profile. Currently, the only tag value you can pass is the constant `kPMSourceProfile`. See "Tag Constants" (page 128) for more information on this constant.

profile

A pointer to a structure of type `CMProfileLocation` that specifies the location of a ColorSync profile. The profile must be version 2 or later. If you pass a profile that is an earlier version, the function returns the result code `kPMNotImplemented`.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

You can use the function `PMSetProfile` to tag QuickDraw drawing with a custom ColorSync profile. The function `PMSetProfile` is useful only if the graphics context is QuickDraw and the current port is the printing port.

You should call this function each time you want to change the profile used to draw page elements. The printing system resets the profile to the default at the beginning of each page. If you call the function `PMSetProfile` a second time, the old profile is ignored.

Special Considerations

This function is deprecated because QuickDraw is deprecated. When drawing with Quartz, the current stroke and fill color space and the color space associated with an image are used to characterize color. Quartz provides ways to use ColorSync profiles to create color spaces, so you can characterize color using ColorSync simply by drawing with Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSetResolution

Sets the application drawing resolution. (Deprecated in Mac OS X v10.5. Draw using Quartz 2D and call `CGContextScaleCTM` instead.)

```
OSStatus PMSetResolution (
   PMPageFormat pageFormat,
   const PMResolution *res
);
```

Parameters

`pageFormat`

The page format object whose drawing resolution you want to set.

`res`

A pointer to a structure of type `PMResolution` (page 111) that specifies the desired drawing resolution for your application. You should specify the best resolution for your data. The printing system handles the mapping between the resolution you specify and the printer resolution.

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Discussion

If you call this function after initiating a print job, the change is ignored for the current job.

Special Considerations

This function was needed in the past because QuickDraw uses integer coordinates and has no notion of scaling coordinate systems. For Quartz drawing, this function is obsolete. To change the resolution, draw with fractional coordinates or scale the coordinate system and draw with integer coordinates.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`PMCoreDeprecated.h`

PMSetUnadjustedPaperRect

Requests a particular paper size, unaffected by rotation, resolution, or scaling. (Deprecated in Mac OS X v10.5. To set a particular paper size, obtain or create a `PMPaper` (page 109) object and call `PMCreatePageFormatWithPMPaper` (page 30).)

```
OSStatus PMSetUnadjustedPaperRect (
   PMPageFormat pageFormat,
   const PMRect *paperRect
);
```

Parameters

pageFormat

The page format object whose unadjusted paper rectangle you want to set.

paperRect

A pointer to a structure of type PMRect that specifies the desired paper size, in points. The coordinates of the upper-left corner of the paper rectangle are specified relative to the page rectangle. See *Supporting Printing in Your Carbon Application* for more information on page and paper rectangles.

Return Value

A result code. See "Core Printing Result Codes" (page 131). The result code kPMValueOutOfRange indicates that the printer driver does not support the requested page size.

Discussion

This function is not recommended. You should call this function only if your application provides desktop publishing and the Page Setup dialog does not provide sufficient control. Typically, such applications display their own specialized document format dialog.

If you decide to use this function, you must call it between the creation and release of a printing session. After using the function PMSetUnadjustedPaperRect you should always call PMSessionValidatePageFormat (page 93) then call PMGetUnadjustedPaperRect (page 41) to verify that the paper size you set is recorded by the printer driver.

If you call this function after initiating a print job, the change is ignored for the current job.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMUnflattenPageFormat

Rebuilds a page format object from a Memory Manager handle that contains flattened page format data. (Deprecated in Mac OS X v10.5. Use PMPageFormatCreateWithDataRepresentation (page 42) instead.)

```
OSStatus PMUnflattenPageFormat (
   Handle flatFormat,
   PMPageFormat *pageFormat
);
```

Parameters

flatFormat

A handle to a previously flattened page format object. You are responsible for disposing of the handle.

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a page format object that contains the data retrieved from the flattened page format data. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131). The result code kPMInvalidParameter is returned if the flattened PMPageFormat object was created by an incompatible version of Core Printing.

Discussion

The PMUnflattenPageFormat function creates a new PMPageFormat object that contains the data from the flattened page format data. You should call the function PMSessionValidatePageFormat (page 93) to make sure the page format object contains valid values.

If the function returns the result code kPMInvalidParameter you need to create a new, default page format object. You should also notify the user that the flattened page format is not valid.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMUnflattenPageFormatWithCFData

Rebuilds a page format object from a Core Foundation data object that contains flattened page format data. (Deprecated in Mac OS X v10.5. Use PMPageFormatCreateWithDataRepresentation (page 42) instead.)

```
OSStatus PMUnflattenPageFormatWithCFData (
   CFDataRef flattenCFData,
   PMPageFormat *pageFormat
);
```

Parameters

flattenCFData

A Core Foundation data object that contains a flattened representation of a page format object.

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a page format object that is rebuilt from the specified Core Foundation data object. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

PMFlattenPageFormatToCFData (page 172)

Declared In

PMCoreDeprecated.h

PMUnflattenPageFormatWithURL

Rebuilds a page format object from a file system URL that contains flattened page format data. (Deprecated in Mac OS X v10.5. Instead read the data into a CFData object and use PMPageFormatCreateWithDataRepresentation (page 42).)

```
OSStatus PMUnflattenPageFormatWithURL (
   CFURLRef flattenFileURL,
   PMPageFormat *pageFormat
);
```

Parameters

flattenFileURL

A Core Foundation URL that specifies a file containing a flattened representation of a page format object.

pageFormat

A pointer to your PMPageFormat (page 108) variable. On return, the variable refers to a page format object that is rebuilt from the specified file. You are responsible for releasing the page format object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

PMFlattenPageFormatToURL (page 172)

Declared In

PMCoreDeprecated.h

PMUnflattenPrintSettings

Rebuilds a print settings object from a Memory Manager handle that contains flattened print settings data. (Deprecated in Mac OS X v10.5. Use PMPrintSettingsCreateWithDataRepresentation (page 72) instead.)

```
OSStatus PMUnflattenPrintSettings (
   Handle flatSettings,
   PMPrintSettings *printSettings
);
```

Parameters

flatSettings

A handle to a flattened representation of a print settings object.

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a print settings object that contains the data retrieved from the flattened print settings. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131). The result code kPMInvalidParameter is returned if the flattened PMPrintSettings object was created by an incompatible version of Core Printing.

Discussion

The PMUnflattenPrintSettings function creates a new PMPrintSettings object containing the data from the flattened print settings. You should call the function PMSessionValidatePrintSettings (page 94), as some values in the print settings object may no longer be valid.

If the function returns the result code kPMInvalidParameter you need to create a new, default print settings object. You should also notify the user that the print settings are not valid.

There are no scoping requirements as to when you may use this function.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

PMCoreDeprecated.h

PMUnflattenPrintSettingsWithCFData

Rebuilds a print settings object from a Core Foundation data object that contains flattened print settings data. (Deprecated in Mac OS X v10.5. Use PMPrintSettingsCreateWithDataRepresentation (page 72) instead.)

```
OSStatus PMUnflattenPrintSettingsWithCFData (
   CFDataRef flattenCFData,
   PMPrintSettings *printSettings
);
```

Parameters

flattenCFData

A flattened representation of a print settings object.

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a print settings object rebuilt from the specified Core Foundation data object. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

PMFlattenPrintSettingsToCFData (page 174)

Declared In

PMCoreDeprecated.h

PMUnflattenPrintSettingsWithURL

Rebuilds a print settings object from a file that contains flattened print settings data. (Deprecated in Mac OS X v10.5. Instead read the data into a CFData object and use PMPrintSettingsCreateWithDataRepresentation (page 72).)

```
OSStatus PMUnflattenPrintSettingsWithURL (
   CFURLRef flattenFileURL,
   PMPrintSettings *printSettings
);
```

Parameters

flattenFileURL

A file containing a flattened representation of a print settings object.

printSettings

A pointer to your PMPrintSettings (page 110) variable. On return, the variable refers to a print settings object rebuilt from the specified file. You are responsible for releasing the print settings object with the function PMRelease (page 76).

Return Value

A result code. See "Core Printing Result Codes" (page 131).

Availability

Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.5.

See Also

PMFlattenPrintSettingsToURL (page 174)

Declared In

PMCoreDeprecated.h

Document Revision History

This table describes the changes to *Core Printing Reference*.

Date	Notes
2007-07-24	New document describing the C API that provides an abstraction layer for application printing.

Index