NSPrintOperation

Inherits From: NSObject

Conforms To: NSObject (NSObject)

Declared In: AppKit/NSPrintOperation.h

Class Description

An NSPrintOperation object controls operations that generate Encapsulated PostScript (EPS) code or PostScript print jobs. Generally, EPS code is used to transfer images between applications, which happens when the user copies and pastes graphics, uses a Service, or uses ObjectLinks. PostScript is generated when the user prints and faxes documents. An NSPrintOperation object does not generate PostScript code itself; it just controls the overall process, relying on an NSView object to generate the actual code.

NSPrintOperation works in conjunction with two other objects: an NSPrintInfo object, which specifies how the code should be generated, and an NSView object, which performs the actual code generation. You specify these two objects in the method you use to create an NSPrintOperation object. If no NSPrintInfo is specified, NSPrintOperation uses the shared NSPrintInfo, which contains default values. (A shared NSPrintInfo object is automatically created for an application.) The shared NSPrintInfo works well for applications that are not document-based. However, document-based applications should create an NSPrintInfo for each document that might be printed or copied and use that object instead. This will allow users to set printing attributes on a per-document basis.

You create NSPrintOperation objects in any method that is invoked when a user chooses a Print or Copy command. That method must also send **runOperation** to the NSPrintOperation object to start the actual operation. For example, applications that are not document-based have a simple **print:** method as in:

```
- (void)print:sender {
    [[NSPrintOperation printOperatationWithView:self] runOperation];
}
```

However, document-based applications should use their own instances of NSPrintInfo as in:

```
- (void)print:sender {
    [[NSPrintOperation printOperationWithView:[self myView] printInfo:[document
       docPrintInfo]] runOperation];
}
```

This method creates an NSPrintOperation for a print job that uses the document's NSPrintInfo object—not the shared NSPrintInfo object.

In both examples, because this is a print job, the NSPrintOperation object will display an NSPrintPanel object allowing the user to select printing options (i.e., number of pages to print and range of pages to print).

The NSPrintOperation object copies the NSPrintInfo object, updates this copy with information from the NSPrintPanel object, and uses the specified NSView to perform the operation. Some of the information stored in an NSPrintInfo object is constant for a particular document, such as its page size. Other information that is likely to change between print operations is set to default values before the operation begins. In this way, even though NSPrintOperation updates the NSPrintInfo with information from the NSPrintPanel for a specific print job, that information is reset back to the default values for each print job. Because NSPrintOperation keeps a copy of the NSPrintInfo it uses, you could duplicate a specific print job by storing and reusing that copy.

When repeating a print job, you can suppress the display of the NSPrintPanel object by sending **setShowPanels:**, passing NO as the argument, to the NSPrintOperation object before sending it **runOperation**. However, make sure that any non-default settings in the NSPrintInfo object that would normally be selected from a NSPrintPanel object are set to reasonable values—a copy of an NSPrintInfo object used in a previous print job will have the correct values.

You can also customize the NSPrintPanel object using the **setAccessoryView:** method or specify your own NSPrintPanel object using **setPrintPanel:** (an OpenStep addition).

If you want the PostScript code generated for EPS and printing to be the same but different from the code generated for the screen, you can test for this case by sending **isDrawingToScreen** to the current NSDPSContext as in:

```
if (![[NSDPSContext currentContext] isDrawingToScreen]){
   /* Insert EPS and printing code here */
}
```

If you want to generate different PostScript code when printing vs. creating EPS, you can test for this case by sending **isEPSOperation** to the current operation as follows:

```
if ((![[NSDPSContext currentContext] isDrawingToScreen] &&
      ![[NSPrintOperation currentOperation] isEPSOperation])){
   /* Insert printing code here */
}
```

Method Types

Creating an NSPrintOperation

+ EPSOperationWithView:insideRect:toData:
+ EPSOperationWithView:insideRect:toData:printInfo:
+ EPSOperationWithView:insideRect:toPath:printInfo:
+ printOperationWithView:
+ printOperationWithView:printInfo:
– initEPSOperationWithView:insideRect:toData:printInfo:
– initWithView:printInfo:

Setting the current NSPrintOperation for this thread	+ currentOperation + setCurrentOperation:
Determining the type of operation	– isEPSOperation
Modifying the NSPrintInfo object	– printInfo – setPrintInfo:
Getting the NSView object	– view
Running a print operation	– runOperation – cleanUpOperation – deliverResult
Modifying the user interface	– showPanels – setShowPanels: – accessoryView – setAccessoryView: – printPanel – setPrintPanel:
Managing the DPS context	– context – createContext – destroyContext
Modifying page information	– currentPage – pageOrder – setpageOrder:

Class Methods

EPSOperationWithView:insideRect:toData:

+ (NSPrintOperation *)**EPSOperationWithView:**(NSView *)*aView* **insideRect:**(NSRect)*rect* **toData:**(NSMutableData *)*data*

Returns a new NSPrintOperation object that controls the copying of EPS graphics from the area specified by *rect* in *aView*. The new NSPrintOperation object will use the default NSPrintInfo object. The EPS code is written to *data*. Raises an NSPrintOperationExistsException if there is already a print operation in progress, otherwise the returned object is made the current print operation for this thread.

See also: + **EPSOperationWithView:insideRect:toData:printInfo:,**
+ **EPSOperationWithView:insideRect:toPath:printInfo:**

EPSOperationWithView:insideRect:toData:printInfo:

+ (NSPrintOperation *)**EPSOperationWithView:**(NSView *)*aView* **insideRect:**(NSRect)*rect*
toData:(NSMutableData *)*data* **printInfo:**(NSPrintInfo *)*aPrintInfo*

Returns a new NSPrintOperation object that controls the copying of EPS graphics from the area specified by *rect* in *aView*. The new NSPrintOperation object will use the settings stored in *aPrintInfo*. The code is written to *data*. Raises an NSPrintOperationExistsException if there is already a print operation in progress, otherwise the returned object is made the current print operation for this thread.

See also: + **EPSOperationWithView:insideRect:toData:,**
+ EPSOperationWithView:insideRect:toPath:printInfo:

EPSOperationWithView:insideRect:toPath:printInfo:

+ (NSPrintOperation *)**EPSOperationWithView:**(NSView *)*aView* **insideRect:**(NSRect)*rect*
toPath:(NSString *)*path* **printInfo:**(NSPrintInfo *)*aPrintInfo*

Creates and returns a new NSPrintOperation object that controls the copying of EPS graphics from the area specified by *rect* in *aView*. The new NSPrintOperation object will use the settings stored in *aPrintInfo*. The code is written to *path*. Raises an NSPrintOperationExistsException if there is already a print operation in progress, otherwise the returned object is made the current print operation for this thread.

See also: + **EPSOperationWithView:insideRect:toData:,**
+ EPSOperationWithView:insideRect:toData:printInfo:

currentOperation

+ (NSPrintOperation *)**currentOperation**

Returns the current print operation for this thread. Returns **nil** if there isn't a current operation.

See also: + **setCurrentOperation:**

printOperationWithView:

+ (NSPrintOperation *)**printOperationWithView:**(NSView *)*aView*

Returns a new NSPrintOperation that controls the printing of *aView*. The new NSPrintOperation object will use the settings stored in the shared NSPrintInfo object. Raises an NSPrintOperationExistsException if there is already a print operation in progress, otherwise the returned object is made the current print operation for this thread.

See also: + **printOperationWithView:printInfo:**

printOperationWithView:printInfo:

+ (NSPrintOperation *)**printOperationWithView:**(NSView *)*aView* **printInfo:**(NSPrintInfo *)*aPrintInfo*

Returns a new NSPrintOperation that controls the printing of *aView*. The new NSPrintOperation object will use the settings stored in *aPrintInfo*. Raises an NSPrintOperationExistsException if there is already a print operation in progress, otherwise the returned object is made the current print operation for this thread.

See also: + **printOperationWithView:**

setCurrentOperation:

+ (void)**setCurrentOperation:**(NSPrintOperation *)*operation*

Sets the current print operation for this thread to *operation*. If operation is **nil**, then there is no current print operation.

See also: + **currentOperation**

Instance Methods

accessoryView

– (NSView *)accessoryView

Returns the accessory view used by the NSPrintPanel object. You use **setAccessoryView:** to customize the default NSPrintPanel object without having to subclass NSPrintPanel or specify your own NSPrintPanel object.

See also: – **printPanel**, – **setPrintPanel:, – setShowPanels:, – showPanels**

cleanUpOperation

– (void)**cleanUpOperation**

Invoked by **runOperation** at the end of an operation to remove the receiver as the current operation. You typically do not invoke this method directly.

context

– (NSDPSContext *)**context**

Returns the receiver's DPS context used for generating output.

See also: – **createContext**, – **destroyContext**

createContext

– (NSDPSContext *)**createContext**

Creates the DPS context for output generation, using the receiver's NSPrintInfo settings. Do not invoke this method directly—it's invoked before any output is generated.

See also: – **context**, – **destroyContext**

currentPage

– (int)**currentPage**

Returns the page number of the page that is currently being printed.

See also: – **pageOrder**, – **setPageOrder:**

deliverResult

– (BOOL)**deliverResult**

Delivers the results generated by **runOperation** to the intended destination (i.e., the printer spool, or preview application). Returns YES if the operation was successful, otherwise NO. Do not invoke this method directly—it's invoked automatically when the operation is done generating the output.

destroyContext

– (void)**destroyContext**

Destroys the receiver's DPS context. Do not invoke this method directly—it's invoked at the end of a print operation.

See also: – **context**, – **createContext**

initEPSOperationWithView:insideRect:toData:printInfo:

– (id)**initEPSOperationWithView:**(NSView *)*aView* **insideRect:**(NSRect)*rect*
toData:(NSMutableData *)*data* **printInfo:**(NSPrintInfo *)*aPrintInfo*

Initializes and returns a newly allocated NSPrintOperation object to control the copying of EPS graphics from the area specified by *rect* in *aView*, using the settings stored in *aPrintInfo*. This method makes a copy of *aPrintInfo* —*aPrintInfo* is not used in the actual operation. The EPS code is written to *data*.

See also: – **initWithView:printInfo:**

initWithView:printInfo:

– (id)**initWithView:**(NSView *)*aView* **printInfo:**(NSPrintInfo *)*aPrintInfo*

Initializes and returns a newly allocated NSPrintOperation object to control the printing of *aView*, using the settings stored in *aPrintInfo*. This method makes a copy of *aPrintInfo* —*aPrintInfo* is not used in the actual operation. This method is the designated initializer for this class.

See also: – **initEPSOperationWithView:insideRect:toData:printInfo:**

isEPSOperation

– (BOOL)**isEPSOperation**

Returns YES if the receiver controls an EPS operation (initiated by a copy command), and NO if the receiver controls a printing operation (initiated by a print command).

pageOrder

– (NSPrintingPageOrder)**pageOrder**

Returns the order in which pages will be printed. See **setPageOrder:** for possible return values.

See also: – **currentPage**

printInfo

– (NSPrintInfo *)**printInfo**

Returns the receiver's NSPrintInfo object.

See also: – **setPrintInfo:**

printPanel

– (NSPrintPanel *)**printPanel**

Returns the NSPrintPanel object used when running the operation.

See also: – **accessoryView**, – **setAccessoryView**:, – **setPrintPanel:, – setShowPanels:, – showPanels**

runOperation

– (BOOL)**runOperation**

Runs the operation (i.e., copys an EPS graphic or prints a job). Returns YES if successful, otherwise NO.

See also: – **cleanUpOperation**, – **deliverResult**

setAccessoryView:

– (void)setAccessoryView:(NSView *)*aView*

Allows you to augment the NSPrintPanel object by adding a custom NSView (by using this method you do not need to subclass NSPrintPanel or specify your own NSPrintPanel object). The NSPrintPanel is automatically resized to accommodate the new accessory view *aView*.

See also: – **accessoryView**, – **printPanel**, – **setPrintPanel:, – setShowPanels:, – showPanels**

setPageOrder:

– (void)**setPageOrder:**(NSPrintingPageOrder)*order*

Sets the order in which pages will be printed to *order* where *order* is one of:

NSAscendingPageOrder	Ascending (back to front) page order.
NSDescendingPageOrder	Descending (front to back) page order.
NSSpecialPageOrder	The spooler will not rearrange pages—they are print in the order received by the spooler.
NSUnknownPageOrder	No page order specified.

See also: – **currentPage**, – **pageOrder**

setPrintInfo:

– (void)**setPrintInfo:**(NSPrintInfo *)*aPrintInfo*

Sets the receiver's NSPrintInfo object to *aPrintInfo*.

See also: – **printInfo**

setPrintPanel:

– (void)**setPrintPanel:**(NSPrintPanel *)*panel*

Sets the receiver's NSPrintPanel used in the operation to *panel*.

See also: – **accessoryView**, – **printPanel**, – **setAccessoryView:, – setShowPanels:, – showPanels**

setShowPanels:

– (void)**setShowPanels:**(BOOL)*flag*

If *flag* is YES then the NSPrintPanel will be used in the operation, otherwise it will not.

See also: – **accessoryView**, – **printPanel**, – **setAccessoryView:, – setPrintPanel:, – showPanels**

showPanels

– (BOOL)**showPanels**

Returns YES if the NSPrintPanel will be used in the operation, otherwise NO.

See also: – **accessoryView**, – **printPanel**, – **setAccessoryView:, – setPrintPanel:, – setShowPanels:**

view

– (NSView *)**view**

Returns the NSView object that generates the actual EPS or PostScript code controlled by the receiver.