3

Display PostScript

Classes

The classes listed here and the protocol in the following section constitute OpenStep's object-oriented interface to the Display PostScript System. As such, many of the argument and return types that appear below (specifically, those having a ªDPSº prefix) are not described in this document. Rather, they are detailed in the specification for the Display PostScript System itself, as found in the *Display PostScript System Reference Manual*, by Adobe Systems, Inc.

NSDPSContext

Inherits From: NSObject

Conforms To: NSCoding
NSObject

Declared In: dpsclient/NSDPSContext.h

Initializing a Context

- **initWithMutableData:**(NSMutableData *)*data*
  forDebugging:(BOOL)*debug*
  languageEncoding:(DPSProgramEncoding)*langEnc*
  nameEncoding:(DPSNameEncoding)*nameEnc*
  textProc:(DPSTextProc)*textProc*
  errorProc:(DPSErrorProc)*errorProc*

  Initializes a newly allocated NSDPSContext that writes its output to *data* using the language and name encodings specified by *langEnc* and *nameEnc*. The callback functions *textProc* and *errorProc* handle text and errors generated by the context. If *debug* is YES, the output is given in human-readable form in which large structures (such as images) may be represented by comments.

Testing the Drawing Destination

- (BOOL)**isDrawingToScreen**

  Returns YES if the drawing destination is the screen.

Accessing Context Data

- (NSMutableData *)**mutableData**

  Returns the receiver's data object.

Setting and Identifying the Current Context

- (DPSContext)**context**

  Returns the corresponding DPScontext.

+ (NSDPSContext *)**currentContext**

  Returns the current context of the current thread.

+ (void)**setCurrentContext:**(NSDPSContext *)*context*
Installs *context* as the current context of the current thread.

Controlling the Context

- (void)**flush** Forces any buffered data to be sent to its destination.
- (void)**interruptExecution** Interrupts execution in the receiver's context.
- (void)**notifyObjectWhenFinishedExecuting:**(id <NSDPSContextNotification>)*object*
Registers *object* to receive a **contextFinishedExecuting**: message when the NSDPSContext's destination is ready to receive more input.
- (void)**resetCommunication** Discards any data that hasn't already been sent to its destination.
- (void)**wait** Waits until the NSDPSContext's destination is ready to receive more input.

Managing Returned Text and Errors

- (DPSErrorProc)**errorProc** Returns the context's error callback function.
- (void)**setErrorProc:**(DPSErrorProc)*proc* Sets the context's error callback function to *proc*.
- (void)**setTextProc:**(DPSTextProc)*proc* Sets the context's text callback function to *proc*.
+ (NSString *)**stringForDPSError:**(const DPSBinObjSeqRec *)*error*
Returns a string representation of *error*.
- (DPSTextProc)**textProc** Returns the context's text callback function.

Sending Raw Data

- (void)**writeData:**(NSData *)*data* Sends the PostScript data in *data* to the context's destination.
- (void)**writePostScriptWithLanguageEncodingConversion:**(NSData *)*data*
Writes the PostScript data in *data* to the context's destination. The data, formatted as plain text, encoded tokens, or a binary object sequence, is converted as necessary depending on the language encoding of the

receiving context.

- (void)**printFormat:**(NSString *)*format,...* — Constructs a string from *format* and following string objects (in the manner of **printf()**) and sends it to the context's destination.

- (void)**printFormat:**(NSString *)*format* **arguments:**(va_list)*argList* — Constructs a string from *format* and *argList* (in the manner of **vprintf()**) and sends it to the context's destination.

Managing Binary Object Sequences

- (void)**awaitReturnValues** — Waits for all return values from the result table.

- (void)**writeBOSArray:**(const void *)*data* **count:**(unsigned int)*count* **ofType:**(DPSDefinedType)*type* — Write an array to the context's destination as part of a a binary object sequence. The array is taken from *data* and consists of *count* items of type *type*.

- (void)**writeBOSNumString:**(const void *)*data* **length:**(unsigned int)*numBytes* **ofType:**(DPSDefinedType)*type* **scale:**(int)*scale* — Write a number string to the context's destination as part of a binary object sequence. The string is taken from *data* as described by *numBytes*, *type,* and *scale.*

- (void)**writeBOSString:**(const void *)*data* **length:**(unsigned int)*numBytes* — Write a string to the context's destination as part of a binary object sequence. The string is taken from *numBytes* of *data.*

- (void)**writeBinaryObjectSequence:**(const void *)*data* **length:**(unsigned int)*numBytes* — Write a binary object sequence to the context's destination. The sequence consists of *numBytes* of *data.*

- (void)**updateNameMap** — Updates the context's name map from the client library's name map.

Managing Chained Contexts

- (void)**chainChildContext:**(NSDPSContext *)*child* — Links *child* (and all of it's children) to the receiver as its chained context, a context that receives a copy of all PostScript code sent to the receiver.

- (NSDPSContext *)**childContext** — Returns the receiver's child context, or **nil** if none exists.

- (NSDPSContext *)**parentContext** — Returns the receiver's parent context, or **nil** if none exists.

- (void)**unchainContext**	Unlinks the child context (and all of it's children) from the receiver's list of chained contexts.

Debugging Aids

+ (BOOL)**areAllContextsOutputTraced**	Returns YES if the data flowing between the application's contexts and their destinations is copied to diagnostic output.
+ (BOOL)**areAllContextsSynchronized**	Returns YES if all NSPDSContext objects invoke the **wait** method after sending each batch of output.
+ (void)**setAllContextsOutputTraced:**(BOOL)*flag*	Causes the data (PostScript code, return values, etc.) flowing between the all the application's contexts and their destinations to be copied to diagnostic output.
+ (void)**setAllContextsSynchronized:**(BOOL)*flag*	Causes the **wait** method to be invoked each time an NSDPSContext object sends a batch of output to its destination.
- (BOOL)**isOutputTraced**	Returns YES if the data flowing between the application's single context and its destination is copied to diagnostic output.
- (BOOL)**isSynchronized**	Returns whether the **wait** method is invoked each time the receiver sends a batch of output to the server.
- (void)**setOutputTraced:**(BOOL)*flag*	Causes the data (PostScript code, return values, etc.) flowing between the application's single context and the window server to be copied to diagnostic output.
- (void)**setSynchronized:**(BOOL)*flag*	Sets whether the **wait** method is invoked each time the receiver sends a batch of output to its destination.