Defined Types

DPSContextRec

DECLARED IN dpsclient/dpsfriends.h

SYNOPSIS

```
typedef struct _t_DPSContextRec {
        char *priv;
        DPSSpace space;
        DPSProgramEncoding programEncoding;
        DPSNameEncoding nameEncoding;
        struct _t_DPSProcsRec const * procs;
        void (*textProc)();
        void (*errorProc)();
        DPSResults resultTable;
        unsigned int resultTableLength;
        struct _t_DPSContextRec *chainParent, *chainChild;
        DPSContextType type;
} DPSContextRec, *DPSContext;
```

DESCRIPTION The **DPSContextRec** structure represents a Display PostScript context.

DPSContextType

DECLARED IN dpsclient/dpsfriends.h

SYNOPSIS

```
typedef enum {
        dps_machServer,
        dps_fdServer,
        dps_stream
} DPSContextType;
```

DESCRIPTION These represent the context types supported by NeXT's version of Display PostScript, as used in the **type** field of a **DPSContextRec** structure.

DPSErrorCode

DECLARED IN dpsclient/dpsclient.h

SYNOPSIS

typedef enum _DPSErrorCode {
    dps_err_ps = DPS_ERROR_BASE,
    dps_err_nameTooLong,
    dps_err_resultTagCheck,
    dps_err_resultTypeCheck,
    dps_err_invalidContext,
    dps_err_select = DPS_NEXT_ERROR_BASE,
    dps_err_connectionClosed,
    dps_err_read,
    dps_err_write,
    dps_err_invalidFD,
    dps_err_invalidTE,
    dps_err_invalidPort,
    dps_err_outOfMemory,
    dps_err_cantConnect
} **DPSErrorCode**;

DESCRIPTION Error codes passed to a **DPSErrorProc()** function.

DPSEventFilterFunc

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS typedef int (***DPSEventFilterFunc**)(NXEvent **ev*);

DESCRIPTION Call-back function used to filter events.

DPSFDProc

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS typedef void (***DPSFDProc**)(int *fd*, void **userData*);

DESCRIPTION Call-back function used when a file descriptor is registered through **DPSAddFD()**.

DPSNumberFormat

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

```
typedef enum _DPSNumberFormat {
#ifdef __BIG_ENDIAN__
        dps_float = 48,
        dps_long = 0,
        dps_short = 32
#else
        dps_float = 48+128,
        dps_long = 0+128,
        dps_short = 32+128
} DPSNumberFormat;
```

DESCRIPTION These constants are used by the **DPSDoUserPath()** function to describe the type of numbers that are being passed.

DPSPingProc

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

```
typedef void (*DPSPingProc)
        (DPSContext ctxt,
        void *userData);
```

DESCRIPTION Call-back function used by **DPSAsynchronousWaitContext()**.

DPSPortProc

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS typedef void (*__DPSPortProc__)
    (msg_header_t **msg*,
    void **userData*);

DESCRIPTION Call-back function used when a port is registered through **DPSAddPort()**.

DPSTimedEntry

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS typedef struct __DPSTimedEntry ***DPSTimedEntry**;

DESCRIPTION The return type for **DPSAddTimedEntry()**.

DPSTimedEntryProc

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS typedef void (*__DPSTimedEntryProc__)
    (DPSTimedEntry *timedEntry*,
    double *now*,
    void **userData*);

DESCRIPTION Call-back function used when a timed entry is registered through **DPSAddTimedEntry()**.

DPSUserPathAction

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

typedef enum _DPSUserPathAction {
    dps_uappend,
    dps_ufill,
    dps_ueofill,
    dps_ustroke,
    dps_ustrokepath,
    dps_inufill,
    dps_inueofill,
    dps_inustroke,
    dps_def,
    dps_put
} **DPSUserPathAction**;

DESCRIPTION These constants are convenient representations of some of the PostScript operator indices, suitable for enrollment in the action array passed to **DPSDoUserPath()**.

DPSUserPathOp

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

typedef enum _DPSUserPathOp {
    dps_setbbox,
    dps_moveto,
    dps_rmoveto,
    dps_lineto,
    dps_rlineto,
    dps_curveto,
    dps_rcurveto,
    dps_arc,
    dps_arcn,
    dps_arct,
    dps_closepath,
    dps_ucache
} **DPSUserPathOp**;

DESCRIPTION These constants represent the PostScript operators that can be passed in **DPSDoUserPath()**'s operator array.

NXCoord

DECLARED IN dpsclient/event.h

SYNOPSIS typedef float **NXCoord**

DESCRIPTION Used to represent a single coordinate in a Cartesian coordinate system.

NXEvent

DECLARED IN dpsclient/event.h

SYNOPSIS

```
typedef struct _NXEvent {
        int type;
        NXPoint location;
        long time;
        int flags;
        unsigned int window;
        NXEventData data;
        DPSContext ctxt;
} NXEvent, *NXEventPtr;
```

DESCRIPTION Represents a single event; this structure is also known as the *event record*. The fields are:

type	The type of event (see “Event Types,” below)
location	The event’s location in the base coordinate system of its window
time	The time of the event (in hardware-dependent units) since system startup
flags	Mouse-button and modifier-key flags (see “Event Flags,” below)
window	The window number of the window associated with the event
data	Additional type-specific data (see “NXEventData,” below)
ctxt	The PostScript context of the event

NXEventData

DECLARED IN dpsclient/event.h

SYNOPSIS

```
typedef union {
        struct {
                short eventNum;
                int click;
                unsigned char pressure;
        } mouse;
        struct {
                short repeat;
                unsigned short charSet;
                unsigned short charCode;
                unsigned short keyCode;
                short keyData;
        } key;
        struct {
                short eventNum;
                int trackingNum;
                int userData;
        } tracking;
        struct {
                short subtype;
                union {
                        float F[2];
                        long L[2];
                        short S[4];
                        char C[8];
                } misc;
        } compound;
} NXEventData;
```

DESCRIPTION This structure supplies type-specific information for an event. It’s a union of four structures, where the type of the event determines which structure is pertinent:

- **mouse** is used for mouse events.
- **key** is used for keyboard events.
- **tracking** is for tracking-rectangle events.
- **compound** is for system-, kit-, and application-defined events.

NXPoint

DECLARED IN dpsclient/event.h

SYNOPSIS

```
typedef struct _NXPoint {
        NXCoord x;
        NXCoord y;
} NXPoint;
```

DESCRIPTION Represents a point in a Cartesian coordinate system.

NXSize

DECLARED IN dpsclient/event.h

SYNOPSIS

```
typedef struct _NXSize {
        NXCoord width;
        NXCoord height;
} NXSize;
```

DESCRIPTION Represents a two-dimensional size.

Symbolic Constants

All Contexts

DECLARED IN	dpsclient/dpsNeXT.h
SYNOPSIS	DPS_ALLCONTEXTS
DESCRIPTION	This constant represents all extant contexts.

Alpha Constants

DECLARED IN	dpsclient/dpsNeXT.h
SYNOPSIS	NX_DATA NX_ONES
DESCRIPTION	These constants represent alpha values.

Character Set Values

DECLARED IN	dpsclient/event.h
SYNOPSIS	NX_ASCIISET NX_SYMBOLSET NX_DINGBATSSET
DESCRIPTION	These constants represent the values that may occur in the **data.key.charSet** field of an NXEvent structure.

Compositing Operations

DECLARED IN	dpsclient/dpsNeXT.h
SYNOPSIS	NX_CLEAR NX_COPY NX_SOVER NX_SIN NX_SOUT NX_SATOP NX_DOVER NX_DIN NX_DOUT NX_DATOP NX_XOR NX_PLUSD NX_HIGHLIGHT NX_PLUSL
DESCRIPTION	These represent the compositing operations used by **PScomposite()** and the NXImage class.

Error Code Bases

DECLARED IN	dpsclient/dpsclient.h
SYNOPSIS	DPS_ERROR_BASE DPS_NEXT_ERROR_BASE
DESCRIPTION	These constants represent the lowest values for Display PostScript error codes.

Event Types

DECLARED IN dpsclient/event.h

SYNOPSIS

Type	Meaning
NX_NULLEVENT	A non-event
NX_LMOUSEDOWN	Left mouse-down
NX_LMOUSEUP	Left mouse-up
NX_LMOUSEDRAGGED	left mouse-dragged
NX_MOUSEDOWN	Same as NX_LMOUSEDOWN
NX_MOUSEUP	Same as NX_LMOUSEUP
NX_MOUSEDRAGGED	Same as NX_LMOUSEDRAGGED
NX_RMOUSEDOWN	Right mouse-down
NX_RMOUSEUP	Right mouse-up
NX_RMOUSEDRAGGED	Right mouse-dragged
NX_MOUSEMOVED	Mouse-moved
NX_MOUSEENTERED	Mouse-entered
NX_MOUSEEXITED	Mouse-exited
NX_KEYDOWN	Key-down
NX_KEYUP	Key-up event
NX_FLAGSCHANGED	Flags-changed
NX_KITDEFINED	Application Kit-defined
NX_SYSDEFINED	System-defined
NX_APPDEFINED	Application-defined
NX_TIMER	Timer used for tracking
NX_CURSORUPDATE	Cursor tracking
NX_JOURNALEVENT	Event used by journaling
NX_FIRSTEVENT	The smallest-valued event constant
NX_LASTEVENT	The greatest-valued event constant
NX_ALLEVENTS	A value that includes all event types

DESCRIPTION These constants represent event types. They're passed as the **type** field of the NXEvent structure that's created when an event occurs.

Event Type Masks

DECLARED IN dpsclient/event.h

SYNOPSIS
NX_NULLEVENTMASK
NX_LMOUSEDOWNMASK
NX_LMOUSEUPMASK
NX_RMOUSEDOWNMASK
NX_RMOUSEUPMASK
NX_MOUSEMOVEDMASK
NX_LMOUSEDRAGGEDMASK
NX_RMOUSEDRAGGEDMASK
NX_MOUSEENTEREDMASK
NX_MOUSEEXITEDMASK
NX_KEYDOWNMASK
NX_KEYUPMASK
NX_FLAGSCHANGEDMASK
NX_KITDEFINEDMASK
NX_APPDEFINEDMASK
NX_SYSDEFINEDMASK
NX_TIMERMASK
NX_CURSORUPDATEMASK
NX_MOUSEDOWNMASK
NX_MOUSEUPMASK
NX_MOUSEDRAGGEDMASK
NX_JOURNALEVENTMASK

DESCRIPTION These masks correspond to the event types defined immediately above. They let you query the **type** field of an NXEvent structure for the existence of a particular event type.

Forever

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS NX_FOREVER

DESCRIPTION A long, long time. Typically used as the timeout argument to **DPSGetEvent()**.

Keyboard State Flags Masks

DECLARED IN dpsclient/event.h

SYNOPSIS

Type	**Meaning**
NX_ALPHASHIFTMASK	Shift lock
NX_SHIFTMASK	Shift key
NX_CONTROLMASK	Control key
NX_ALTERNATEMASK	Alt key
NX_COMMANDMASK	Command key
NX_NUMERICPADMASK	Number pad key
NX_HELPMASK	Help key
NX_NEXTCTRLKEYMASK	Control key
NX_NEXTLSHIFTKEYMASK	Left shift key
NX_NEXTRSHIFTKEYMASK	Right shift key
NX_NEXTLCMDKEYMASK	Left command key
NX_NEXTRCMDKEYMASK	Right command key
NX_NEXTLALTKEYMASK	Left alt key
NX_NEXTRALTKEYMASK	Right alt key

DESCRIPTION These masks correspond to keyboard states that might be included in an NXEvent structure's **flags** mask. The masks are grouped as device-independent (NX_ALPHASHIFTMASK through NX_HELPMASK) and device-dependent (all others).

Miscellaneous Event Flags Masks

DECLARED IN dpsclient/event.h

SYNOPSIS

Type	**Meaning**
NX_STYLUSPROXIMITYMASK	Stylus is in proximity (for tablets)
NX_NONCOALSESCEDMASK	Event coalescing disabled

DESCRIPTION These masks correspond to miscellaneous states that might be included in an NXEvent structure's **flags** mask.

Window Backing Types

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

NX_RETAINED
NX_NONRETAINED
NX_BUFFERED

DESCRIPTION These represent the three backing types provided by window devices (and used by the Application Kit's Window objects).

Window Screen List Placement

DECLARED IN dpsclient/dpsNeXT.h

SYNOPSIS

NX_ABOVE
NX_BELOW
NX_OUT

DESCRIPTION These represent the placement of a window device in the screen list.