

**NAME**

**libxdot** – parsing and deparsing of xdot operations

**SYNOPSIS**

```
#include <graphviz/xdot.h>
```

```
typedef enum {
    xd_none,
    xd_linear,
    xd_radial
} xdot_grad_type;
```

```
typedef struct {
    float frac;
    char* color;
} xdot_color_stop;
```

```
typedef struct {
    double x0, y0;
    double x1, y1;
    int n_stops;
    xdot_color_stop* stops;
} xdot_linear_grad;
```

```
typedef struct {
    double x0, y0, r0;
    double x1, y1, r1;
    int n_stops;
    xdot_color_stop* stops;
} xdot_radial_grad;
```

```
typedef struct {
    xdot_grad_type type;
    union {
        char* clr;
        xdot_linear_grad ling;
        xdot_radial_grad ring;
    } u;
} xdot_color;
```

```
typedef enum {
    xd_left, xd_center, xd_right
} xdot_align;
```

```
typedef struct {
    double x, y, z;
} xdot_point;
```

```
typedef struct {
    double x, y, w, h;
} xdot_rect;
```

```
typedef struct {
    int cnt;
```

```

    xdot_point* pts;
} xdot_polyline;

typedef struct {
    double x, y;
    xdot_align align;
    double width;
    char* text;
} xdot_text;

typedef struct {
    xdot_rect pos;
    char* name;
} xdot_image;

typedef struct {
    double size;
    char* name;
} xdot_font;

typedef enum {
    xd_filled_ellipse, xd_unfilled_ellipse,
    xd_filled_polygon, xd_unfilled_polygon,
    xd_filled_bezier, xd_unfilled_bezier,
    xd_polyline,    xd_text,
    xd_fill_color,  xd_pen_color, xd_font, xd_style, xd_image,
    xd_grad_fill_color,  xd_grad_pen_color,
    xd_fontchar
} xdot_kind;

typedef enum {
    xop_ellipse,
    xop_polygon,
    xop_bezier,
    xop_polyline,    xop_text,
    xop_fill_color,  xop_pen_color, xop_font, xop_style, xop_image,
    xop_grad_color,
    xop_fontchar
} xop_kind;

typedef struct _xdot_op xdot_op;
typedef void (*drawfunc_t)(xdot_op*, int);
typedef void (*freefunc_t)(xdot_op*);

struct _xdot_op {
    xdot_kind kind;
    union {
        xdot_rect ellipse; /* xd_filled_ellipse, xd_unfilled_ellipse */
        xdot_polyline polygon; /* xd_filled_polygon, xd_unfilled_polygon */
        xdot_polyline polyline; /* xd_polyline */
        xdot_polyline bezier; /* xd_filled_bezier, xd_unfilled_bezier */
        xdot_text text; /* xd_text */
        xdot_image image; /* xd_image */
    }
};

```

```

    char* color;          /* xd_fill_color, xd_pen_color */
    xdot_color grad_color; /* xd_grad_fill_color, xd_grad_pen_color */
    xdot_font font;       /* xd_font */
    char* style;          /* xd_style */
    unsigned int fontchar; /* xd_fontchar */
} u;
drawfunc_t drawfunc;
};

#define XDOT_PARSE_ERROR 1

typedef struct {
    int cnt;
    int sz;
    xdot_op* ops;
    freefunc_t freefunc;
    int flags;
} xdot;

typedef struct {
    int cnt; /* no. of xdot ops */
    int n_ellipse;
    int n_polygon;
    int n_polygon_pts;
    int n_polyline;
    int n_polyline_pts;
    int n_bezier;
    int n_bezier_pts;
    int n_text;
    int n_font;
    int n_style;
    int n_color;
    int n_image;
    int n_gradcolor;
    int n_fontchar;
} xdot_stats;

xdot* parseXDotF (char*, drawfunc_t opfns[], int sz);
xdot* parseXDotFOn (char*, drawfunc_t opfns[], int sz, xdot*);
xdot* parseXDot (char*);
char* sprintXDot (xdot*);
void fprintfXDot (FILE*, xdot*);
void jsonXDot (FILE*, xdot*);
void freeXDot (xdot*);
int statXDot (xdot*, xdot_stats*);

xdot_grad_type colorType (char*);
xdot_color* parseXDotColor (char*);
void freeXDotColor (xdot_color*);

```

## DESCRIPTION

*libxdot* provides support for parsing and deparsing graphical operations specified by the *xdot* language.

## Types

### **xdot**

This encapsulates a series of *cnt* xdot operations, stored in the array pointed to by *ops*. The *sz* indicates the size of each item stored in *ops*. If the user sets the *freefunc* field, this function will be called on each item in *ops* during *freeXDot* before the library does its own clean up of the item. This allows the user to free any resources stored in the item by using an expansion of the *xdot\_op* structure.

### **xdot\_op**

A value of this type represents one xdot operation. The operation is specified by the *kind* field. The corresponding data is stored in the union *u*, with the subfield associated with a given *kind* indicated by the comments.

The *drawfunc* field allows the user to attach a drawing-specific function to the operation, providing an object-based interface. These functions can be automatically attached during parsing by providing a non-NULL second argument to **parseXDotF**.

### **xop\_kind**

This type provides an enumeration of the allowed xdot operations. See

<https://graphviz.org/docs/outputs/canon/#xdot>  
for the specific semantics associated with each operation.

### **xdot\_rect**

This represents a rectangle. For ellipses, the *x* and *x* fields represent the center of the rectangle, and *w* and *h* give the half-width and half-height, respectively. For images, (*x,y*) gives the lower left corner of the rectangle, and *w* and *h* give the width and height, respectively.

### **xdot\_polyline**

This type encapsulates a series of *cnt* points.

### **xdot\_text**

A value of this type corresponds to printing the string *text* using the baseline point (*x,y*). The *width* field gives an approximation of how wide the printed string will be using the current font and font size. The *align* field indicates how the text should be horizontally aligned with the point (*x,y*).

### **xdot\_image**

This denotes the insertion of an image. The image source is given by *name*. The images is to be placed into the rectangle *pos*.

### **xdot\_font**

The fields give the name and size, in points, of a font.

### **xdot\_align**

This enumeration type corresponds to the xdot alignment values -1, 0 and 1 used with the text operator, or '\l', '\n' and '\r' used in dot text.

## Functions

### **xdot\* parseXDotF (char \*str, drawfunc\_t\* opfns, int sz)**

Parses the string *str* as a sequence of xdot operations and returns a pointer to the resulting *xdot* structure. The function parses as many xdot operations as it can. If some unknown or incorrect input was encountered in *str*, the *ops* and *cnt* fields will reflect the operations parsed before the error, and the *XDOT\_PARSE\_ERROR* bit will be set in the *flags* field. The function returns NULL if it cannot parse anything.

If *sz* is non-zero, it is assumed to be the size of some structure type containing *xdot\_op* as a prefix. In this case, the elements in the array pointed to by *ops* will each have size *sz*.

If *opfns* is non-zero, it is taken to be any array of functions indexed by *xop\_kind*. During parsing, the *drawfunc* member of *xdot\_op* will be set to the corresponding function in *opfns*.

### **xdot\* parseXDotFOn (char \*str, drawfunc\_t\* opfns, int sz, xdot\* x)**

The same as *parseXDotF*, but append to the given *xdot* object *x*.

**xdot\* parseXDot (char \*str)**

This is equivalent to *parseXDotF(str, 0, 0)*.

**void freeXDot (xdot\* xp)**

This frees the resources associated with the argument. If *xp* is NULL, nothing happens.

**extern char\* sprintXDot (xdot\* xp)****extern void fprintfXDot (FILE\* fp, xdot\* xp)**

These two functions deparse the argument *xdot* structure, producing a string representation. *fprintfXDot* writes the output onto the open stream *fp*; *sprintXDot* returns a heap-allocated string.

The color string with fill and draw operations can encode linear and radial gradients. These values are parsed automatically by **parseXDotF** or **parseXDot**, with *xdot\_op* having kind *xd\_grad\_pen\_color* or *xd\_grad\_fill\_color* and the value is stored in *grad\_color*.

For an application that handles its own parsing of *xdot*, the library provides three helper functions.

**void jsonXDot(FILE \*fp, xdot \*x)**

Translate a given *xdot* object to a JSON representation. This functionality is currently considered experimental and the format of the JSON output may not be stable across Graphviz releases.

**xdot\_grad\_type colorTypeXDot (char \*str)**

returns the color type described by the input string.

**char\* parseXDotColor (char \*str, xdot\_color\* clr)**

attempts to parse the string *str* as a color value, storing the result in *clr*. It returns NULL on failure.

**void freeXDotColor (xdot\_color\* cp)**

This frees the resources associated with a value of type *xdot\_color*.

**int statXDot (xdot \*x, xdot\_stats \*sp)**

This function is provided for retrieving various statistics about an *xdot* object. Returns 0 on success and populates the output parameter *sp* with counts of various entities in the *xdot* object.

**BUGS**

Although some small checking is done on the *sz* argument to *parseXDotF*, it is assumed it is a valid value from *sizeof* applied to some structure type containing *xdot\_op* as its first field. There can be no validation of the *opfn*s argument.

**AUTHORS**

Emden R. Gansner (erg@research.att.com).