

**NAME**

XStoreBytes, XStoreBuffer, XFetchBytes, XFetchBuffer, XRotateBuffers - manipulate cut and paste buffers

**SYNTAX**

```
int XStoreBytes(Display *display, _Xconst char *bytes, int nbytes);
```

```
int XStoreBuffer(Display *display, _Xconst char *bytes, int nbytes, int buffer);
```

```
char *XFetchBytes(Display *display, int *nbytes_return);
```

```
char *XFetchBuffer(Display *display, int *nbytes_return, int buffer);
```

```
int XRotateBuffers(Display *display, int rotate);
```

**ARGUMENTS**

*buffer* Specifies the buffer in which you want to store the bytes or from which you want the stored data returned.

*bytes* Specifies the bytes, which are not necessarily ASCII or null-terminated.

*display* Specifies the connection to the X server.

*nbytes* Specifies the number of bytes to be stored.

*nbytes\_return* Returns the number of bytes in the buffer.

*rotate* Specifies how much to rotate the cut buffers.

**DESCRIPTION**

The data can have embedded null characters and need not be null-terminated. The cut buffer's contents can be retrieved later by any client calling **XFetchBytes**.

**XStoreBytes** can generate a **BadAlloc** error.

If an invalid buffer is specified, the call has no effect. The data can have embedded null characters and need not be null-terminated.

**XStoreBuffer** can generate a **BadAlloc** error.

The **XFetchBytes** function returns the number of bytes in the `nbytes_return` argument, if the buffer contains data. Otherwise, the function returns NULL and sets `nbytes` to 0. The appropriate amount of storage is allocated and the pointer returned. The client must free this storage when finished with it by calling **XFree**.

The **XFetchBuffer** function returns zero to the `nbytes_return` argument if there is no data in the buffer or if an invalid buffer is specified.

**XFetchBuffer** can generate a **BadValue** error.

The **XRotateBuffers** function rotates the cut buffers, such that buffer 0 becomes buffer `n`, buffer 1 becomes  $n + 1 \bmod 8$ , and so on. This cut buffer numbering is global to the display. Note that **XRotateBuffers** generates **BadMatch** errors if any of the eight buffers have not been created.

**XRotateBuffers** can generate a **BadMatch** error.

## DIAGNOSTICS

**BadAlloc** The server failed to allocate the requested resource or server memory.

**BadAtom** A value for an Atom argument does not name a defined Atom.

**BadMatch** Some argument or pair of arguments has the correct type and range but fails to match in some other way required by the request.

**BadValue** Some numeric value falls outside the range of values accepted by the request. Unless a specific range is specified for an argument, the full range defined by the argument's type is accepted. Any argument defined as a set of alternatives can generate this error.

## SEE ALSO

XFree(3)

*Xlib - C Language X Interface*