

NAME

Xau library: XauFileName, XauReadAuth, XauLockAuth, XauUnlockAuth, XauWriteAuth, XauDisposeAuth, XauGetAuthByAddr, XauGetBestAuthByAddr - X authority database routines

SYNOPSIS

```
#include <X11/Xauth.h>
```

```
typedef struct xauth {  
    unsigned short family;  
    unsigned short address_length;  
    char *address;  
    unsigned short number_length;  
    char *number;  
    unsigned short name_length;  
    char *name;  
    unsigned short data_length;  
    char *data;  
} Xauth;
```

```
char *XauFileName (void);
```

```
Xauth *XauReadAuth (FILE *auth_file);
```

```
int XauWriteAuth (FILE *auth_file, Xauth *auth);
```

```
Xauth *XauGetAuthByAddr (unsigned short family, unsigned short  
    address_length, const char *address, unsigned short  
    number_length, const char *number, unsigned short  
    name_length, const char *name);
```

```
Xauth *XauGetBestAuthByAddr (unsigned short family, unsigned short  
    address_length, const char *address, unsigned short  
    number_length, const char *number, int types_length,  
    char **types, const int *type_lengths);
```

```
int XauLockAuth (const char *file_name, int retries, int  
    timeout, long dead);
```

```
int XauUnlockAuth (const char *file_name);
```

```
int XauDisposeAuth (Xauth *auth);
```

DESCRIPTION

XauFileName generates the default authorization file name by first checking the XAUTHORITY environment variable if set, else it returns \$HOME/.Xauthority. This name is statically allocated and should not be freed.

XauReadAuth reads the next entry from *auth_file*. The entry is **not** statically allocated and should be freed by calling *XauDisposeAuth*.

XauWriteAuth writes an authorization entry to *auth_file*. It returns 1 on success, 0 on failure.

XauGetAuthByAddr searches for an entry which matches the given network address/display number pair. The entry is **not** statically allocated and should be freed by calling *XauDisposeAuth*.

XauGetBestAuthByAddr is similar to **XauGetAuthByAddr**, except that a list of acceptable authentication methods is specified. Xau will choose the file entry which matches the earliest entry in this list (e.g., the most secure authentication method). The *types* argument is an array of strings, one string for each authentication method. *types_length* specifies how many elements are in the *types* array. *types_lengths* is an array of integers representing the length of each string.

XauLockAuth does the work necessary to synchronously update an authorization file. First it makes two file names, one with “-c” appended to *file_name*, the other with “-l” appended. If the “-c” file already exists and is more than *dead* seconds old, *XauLockAuth* removes it and the associated “-l” file. To prevent possible synchronization troubles with NFS, a *dead* value of zero forces the files to be removed. *XauLockAuth* makes *retries* attempts to create and link the file names, pausing *timeout* seconds between each attempt. *XauLockAuth* returns a collection of values depending on the results:

LOCK_ERROR

A system error occurred, either a *file_name* which is too long, or an unexpected failure from a system call. *errno* may prove useful.

LOCK_TIMEOUT

retries attempts failed

LOCK_SUCCESS

The lock succeeded.

XauUnlockAuth undoes the work of *XauLockAuth* by unlinking both the “-c” and “-l” file names.

XauDisposeAuth frees storage allocated to hold an authorization entry.

SEE ALSO

xauth(1), xdm(1)

AUTHOR

Keith Packard, MIT X Consortium