XrmInitialize, XrmParseCommand, XrmValue, XrmOptionKind, XrmOptionDescRec − initialize the Resource Manager, Resource Manager structures, and parse the command line

void XrmInitialize( );

void XrmParseCommand(*database*, *table*, *table_count*, *name*, *argc_in_out*, *argv_in_out*)
    XrmDatabase **database*;
    XrmOptionDescList *table*;
    int *table_count*;
    char **name*;
    int **argc_in_out*;
    char ***argv_in_out*;

argc_in_out	Specifies the number of arguments and returns the number of remaining arguments.
argv_in_out	Specifies the command line arguments and returns the remaining arguments.
database	Specifies the resource database.
name	Specifies the application name.
table	Specifies the table of command line arguments to be parsed.
table_count	Specifies the number of entries in the table.

The XrmInitialize function initialize the resource manager. It must be called before any other Xrm functions are used.

The **XrmParseCommand** function parses an (argc, argv) pair according to the specified option table, loads recognized options into the specified database with type ''String,'' and modifies the (argc, argv) pair to remove all recognized options. If database contains NULL, **XrmParseCommand** creates a new database and returns a pointer to it. Otherwise, entries are added to the database specified. If a database is created, it is created in the current locale.

The specified table is used to parse the command line. Recognized options in the table are removed from argv, and entries are added to the specified resource database in the order they occur in argv. The table entries contain information on the option string, the option name, the style of option, and a value to provide if the option kind is **XrmoptionNoArg**. The option names are compared byte-for-byte to arguments in argv, independent of any locale. The resource values given in the table are stored in the resource database without modification. All resource database entries are created using a ''String'' representation type. The argc argument specifies the number of arguments in argv and is set on return to the remaining number of arguments that were not parsed. The name argument should be the name of your application for use in building the database entry. The name argument is prefixed to the resourceName in the option table before storing a database entry. The name argument is treated as a single component, even if it has embedded periods. No separating (binding) character is inserted, so the table must contain either a period (.) or an asterisk (*) as the first character in each resourceName entry. To specify a more completely qualified resource name, the resourceName entry can contain multiple components. If the name argument and the resourceNames are not in the Host Portable Character Encoding, the result is implementation-dependent.

The XrmValue, **XrmOptionKind**, and **XrmOptionDescRec** structures contain:

typedef struct {
        unsigned int size;
        XPointer addr;
} XrmValue, *XrmValuePtr;

typedef enum {
        XrmoptionNoArg,   /* Value is specified in XrmOptionDescRec.value */
        XrmoptionIsArg,   /* Value is the option string itself */
        XrmoptionStickyArg,   /* Value is characters immediately following option */
        XrmoptionSepArg,   /* Value is next argument in argv */

```
        XrmoptionResArg,              /* Resource and value in next argument in argv */
        XrmoptionSkipArg,             /* Ignore this option and the next argument in argv */
        XrmoptionSkipLine,            /* Ignore this option and the rest of argv */
        XrmoptionSkipNArgs            /* Ignore this option and the next
                                         XrmOptionDescRec.value arguments in argv */
} XrmOptionKind;

typedef struct {
        char *option;                 /* Option specification string in argv    */
        char *specifier;              /* Binding and resource name (sans application name)    */
        XrmOptionKind argKind;        /* Which style of option it is    */
        XPointer value;               /* Value to provide if XrmoptionNoArg or
                                         XrmoptionSkipNArgs   */
} XrmOptionDescRec, *XrmOptionDescList;
```

**XrmGetResource(3X11), XrmMergeDatabases(3X11), XrmPutResource(3X11),
XrmUniqueQuark(3X11)**
Xlib − C Language X Interface