icon

COLLABORATORS			
	TITLE : icon		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		March 28, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

icon

1.1 icon.doc

```
 AddFreeList()       FreeDiskObject()      MatchToolValue()
 BumpRevision()      FreeFreeList()        PutDiskObject()
 FindToolType()      GetDiskObject()
```

1.2 icon.library/AddFreeList

```
NAME
    AddFreeList - add memory to the free list

SYNOPSIS
    status = AddFreeList( free, mem, len )
    D0                    A0    A1    A2

FUNCTION
    This routine adds the specified memory to the free list.
    The free list will be extended (if required).  If there
    is not enough memory to complete the call, a null is returned.

    Note that AddFreeList does NOT allocate the requested memory.
    It only records the memory in the free list.

INPUTS
    free -- a pointer to a FreeList structure
    mem -- the base of the memory to be recorded
    len -- the length of the memory to be recorded

RESULTS
    status -- nonzero if the call succeeded.

EXCEPTIONS

SEE ALSO
    AllocEntry, FreeEntry, FreeFreeList

BUGS
```

1.3 icon.library/BumpRevision

```
NAME
    BumpRevision - reformat a name for a second copy

SYNOPSIS
    result = BumpRevision( newbuf, oldname  )
    D0                     A0      A1

FUNCTION
    BumpRevision takes a name an turns it into a "copy of name".
    It knows how to deal with copies of copies.  The routine
    will truncate the new name to the maximum dos name size
    (currently 30 characters).

INPUTS
    newbuf - the new buffer that will receive the name (it must
            be at least 31 characters long).
    oldname - the original name

RESULTS
    result - a pointer to newbuf

EXCEPTIONS

EXAMPLE
    oldname                           newbuf
    -------                           ------
    "foo"                             "copy of foo"
    "copy of foo"                     "copy 2 of foo"
    "copy 2 of foo"                   "copy 3 of foo"
    "copy 199 of foo"                 "copy 200 of foo"
    "copy foo"                        "copy of copy foo"
    "copy 0 of foo"                   "copy 1 of foo"
    "012345678901234567890123456789" "copy of 0123456789012345678901"

SEE ALSO

BUGS
```

1.4 icon.library/FindToolType

```
NAME
    FindToolType - find the value of a ToolType variable

SYNOPSIS
    value = FindToolType( toolTypeArray, typeName  )
    D0                    A0             A1

FUNCTION
    This function searches a tool type array for a given entry,
    and returns a pointer to that entry.  This is useful for
    finding standard tool type variables.  The returned
    value is not a new copy of the string but is only
    a pointer to the part of the string after typeName.

INPUTS
    toolTypeArray - an array of strings
    typeName - the name of the tooltype entry

RESULTS
    value - a pointer to a string that is the value bound to
        typeName, or NULL if typeName is not in
        the toolTypeArray.

EXCEPTIONS

EXAMPLE
    Assume the tool type array has two strings in it:
        "FILETYPE=text"
        "TEMPDIR=:t"

    FindToolType( toolTypeArray, "FILETYPE" ) returns "text"
    FindToolType( toolTypeArray, "TEMPDIR" )  returns ":t"
    FindToolType( toolTypeArray, "MAXSIZE" )  returns NULL

SEE ALSO
    MatchToolValue

BUGS
```

1.5 icon.library/FreeDiskObject

```
NAME
    FreeDiskObject - free all memory in a Workbench disk object

SYNOPSIS
    FreeDiskObject( diskobj )
                    A0

FUNCTION
    This routine frees all memory in a Workbench disk object, and the
    object itself.  It is implemented via FreeFreeList().

    GetDiskObject() takes care of all the initialization required
    to set up the objects free list.  This procedure may ONLY
    be called on DiskObject allocated via GetDiskObject().

INPUTS
    diskobj -- a pointer to a DiskObject structure

RESULTS

EXCEPTIONS

SEE ALSO
    GetDiskObject, FreeFreeList
```

```
BUGS
```

1.6 icon.library/FreeFreeList

```
NAME
    FreeFreeList - free all memory in a free list

SYNOPSIS
    FreeFreeList( free )
                  A0

FUNCTION
    This routine frees all memory in a free list, and the
    free list itself.  It is useful for easily getting
    rid of all memory in a series of structures.  There is
    a free list in a Workbench object, and this contains
    all the memory associated with that object.

    A FreeList is a list of MemList structures.  See the
    MemList and MemEntry documentation for more information.

    If the FreeList itself is in the free list, it must be
    in the first MemList in the FreeList.

INPUTS
    free -- a pointer to a FreeList structure

RESULTS

EXCEPTIONS

SEE ALSO
    AllocEntry, FreeEntry, AddFreeList

BUGS
```

1.7 icon.library/GetDiskObject

```
NAME
    GetDiskObject - read in a Workbench disk object

SYNOPSIS
    diskobj = GetDiskObject( name )
    D0                       A0

FUNCTION
    This routine reads in a Workbench disk object in from disk.  The
    name parameter will have a ".info" postpended to it, and the
    info file of that name will be read.  If the call fails,
    it will return zero.  The reason for the failure may be obtained
    via IoErr().
    Using this routine protects you from any future changes to
    the way icons are stored within the system.

    A FreeList structure is allocated just after the DiskObject
    structure; FreeDiskObject makes use of this to get rid of the
    memory that was allocated.

INPUTS
    name -- name of the object

RESULTS
    diskobj -- the Workbench disk object in question

EXCEPTIONS

SEE ALSO
    FreeDiskObject

BUGS
```

1.8 icon.library/MatchToolValue

```
NAME
    MatchToolValue - check a tool type variable for a particular value

SYNOPSIS
    result = MatchToolValue( typeString, value  )
    D0                       A0          A1

FUNCTION
    MatchToolValue is useful for parsing a tool type value for
    a known value.  It knows how to parse the syntax for a tool
    type value (in particular, it knows that '|' separates
    alternate values).

INPUTS
    typeString - a ToolType value (as returned by FindToolType)
    value - you are interested if value appears in typeString

RESULTS
    result - a one if the value was in typeString

EXCEPTIONS

EXAMPLE
    Assume there are two type strings:
        type1 = "text"
        type2 = "a|b|c"

    MatchToolValue( type1, "text" ) returns 1
    MatchToolValue( type1, "data" ) returns 0
    MatchToolValue( type2, "a" ) returns 1
    MatchToolValue( type2, "b" ) returns 1
    MatchToolValue( type2, "d" ) returns 0
    MatchToolValue( type2, "a|b" ) returns 0
```

```
SEE ALSO
    FindToolType

BUGS
```

1.9 icon.library/PutDiskObject

```
NAME
    PutDiskObject - write out a DiskObject to disk

SYNOPSIS
    status = PutDiskObject( name, diskobj )
    D0                      A0    A1

FUNCTION
    This routine writes out a DiskObject structure, and its
    associated information.  The file name of the info
    file will be the name parameter with a
    ".info" postpended to it.  If the call fails, a zero will
    be returned.  The reason for the failure may be obtained
    via IoErr().

    Using this routine protects you from any future changes to
    the way icons are stored within the system.

INPUTS
    name -- name of the object
    diskobj -- a pointer to a DiskObject

RESULTS
    status -- non-zero if the call succeeded

EXCEPTIONS

SEE ALSO
    GetDiskObject, FreeDiskObject

BUGS
```