Example

```
- dms.tovector("1 2.2 3", 'double')     # Vector of strings
[1 2.2 3]
- dms.tovector('1 2.2 3', 'double')     # String (space delimiters)
[1 2.2 3]
- dms.tovector('1,  2.2,  3', 'double') # String (space and comma delimiters)
[1 2.2 3]
- dms.tovector([1,2.2,3], 'float')      # Numeric vector
[1 2.2 3]
- dms.tovector([1,2.2,3], 'integer')    # Float -> integer
- r := [=]
- r.a := '1'
- r.b := unset                          # Unset field
- r.c := 2
- dms.tovector(r,'float', 20)           # Record -> float
[1 20 2]
```

1.6.2 os - Tool
Package utility
Module misc

Functions to perform system-specific operations
include "os.g"

Constructors

Functions

basename	get file part in path names
copy	copy a file, directory, or symlink
dir	return a vector of strings with directory listing
dirname	get directory part in path names
done	end the os tool
edit	edit a file
fileexists	test if files exists
filetime	return the file time
filetimestring	return the file time string
filetype	determine file type
freespace	determine freespace on file systems

fullname	expand path names
isvalidpathname	test if path names are valid
lockinfo	get info about table use and locking
ls	directory listing
mail	mail a message to a user
mkdir	create directories
move	move a file, directory, or symlink
remove	remove files, directories, or symlinks
showtableuse	tell if a table is used or locked
size	determine file or directory sizes
type	Return the type of this tool

Description
The os tool is used to perform system-specific operations, in particular operations on files — for example:

- testing if a file exists
- getting a directory listing
- getting the size of file(s)
- getting the free space on a device
- dealing with path names
- copying, moving, or removing a file or directory

Most functions have a switch telling if a symbolic link has to be followed.
When a path name is given, it can contain environment variables (e.g. `$HOME`) and/or tildes (e.g. ~ or ~`user`).
Note that glish has some built-in functions to do file IO. It also has the `stat` function to obtain file information. However, those functions do not handle environment variables and tilde expansion.

Example

```
include 'os.g'
dos.fileexists ('~/file.name');
dos.freespace ('.');
```

os.isvalidpathname - Function

Package	**utility**
Module	**misc**
Tool	**os**

test if path names are valid

Synopsis
isvalidpathname(pathname)

Description
This function tests if the given path names are valid. It tests if files with the given names exists or can be created.

Arguments

pathname	in	path names Allowed:vector of strings

Returns
vector of bools

---

os.fileexists - Function

Package	**utility**
Module	**misc**
Tool	**os**

test if files exists

Synopsis
fileexists(file, follow)

Description
This function tests if the given files exists. If `follow==T` symbolic links will be followed.

Arguments

file	in	file names Allowed:vector of strings
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of bools

---

os.dir - Function

Package	**utility**
Module	**misc**
Tool	**os**

return a vector of strings with directory listing

Synopsis
dir(directoryname, pattern, types, all, follow)

Description
Get a vector of strings containing the names of files in the given directory. Other arguments can be given to select specific files. To simply list the files in a human-readable format, use dos.ls() instead.

`pattern` can be given to select specific file names using a file name pattern as given in a shell. E.g. `pattern='*.g'` to get all .g files.
By default no name selection is done.

`types` can be given to select on a file (access) types. It is a string containing zero or more letters indicating the required type(s). There are two groups of types:

- `r d s` indicate regular file, directory, resp. symbolic link.
- `R W X` indicate readable, writable, resp. executable.

If one or more letters from a group are given, only such files will be selected.
By default no type selection is done.

`all` can be given to indicate that file names starting with a . should be selected.
By default they are not selected.
Note that even if a pattern `'.*'` is given, `all=T` should be given to get those files.

If `follow==T` symbolic links will be followed. In that case using 's' in the `types` argument makes no sense.

Arguments

directoryname	in	directory name Allowed:string Default: '.'
pattern	in	file name pattern Allowed:string Default: ”
types	in	which types? Allowed:string Default: ”
all	in	also dot files? Allowed:bool Default: F
follow	in	follow symlink? Allowed:bool Default: T

Example

```
dos.dir (pattern='*.pg')   #get all files ending in .pg
dos.dir (types='rX')       #get all regular executable files
v:=dos.dir ('~/aips++/test', types='d')    # get directories
for (nm in v) {                            # (potential tables)
  if (dos.filetype(spaste('~/aips++/test/',nm)) == 'Table') {
    print nm;                                       #show tables
  }
}
```

Returns
vector of strings or fail

---

os.ls - Function

Package	**utility**
Module	**misc**
Tool	**os**

directory listing

Synopsis

ls(dir)

Description

This function lists directories in a human-readable format, like the unix command ls. To retrieve a glish record containing files in a directory, use dos.dir() instead.

Arguments

dir	in	directory name Allowed:string Default: current directory

Example

```
- dos.ls()
aips++.inputsv2.table mymsgs                 tkeep3.g
aips++.records.table  tkeep.g
junk_SP               tkeep2.g
T
- dos.ls('/')
aips++     dev        home       lost+found opt        sbin       users
bin        etc        initrd     misc       proc       tftpboot   usr
boot       export     lib        mnt        root       tmp        var
T
-
```

Returns

T or F

---

os.mkdir - Function

Package	**utility**
Module	**misc**
Tool	**os**

create directories

Synopsis

mkdir(directoryname, makeparent)

Description
Create directories. An error is given if a directory already exists. By default parent directories are not created if they don't exist yet.

Arguments

directoryname	in	directory names Allowed:vector of strings
makeparent	in	make parent directories if needed Allowed:bool Default: F

Returns
T or fail

---

os.fullname - Function

Package	**utility**
Module	**misc**
Tool	**os**

expand path names

Synopsis
fullname(pathname)

Description
Expand path names to their full absolute names by substituting environment variables (e.g. `$HOME`), applying tilde expansion, and prefixing a relative name with the working directory. It also takes care that parts . and .. are removed and that multiple slashes are replaced by a single slash. A possible trailing slash is also removed.
Note that it does not substitute symbolic links.

Example

```
  dos.fullname ('~mhaller')
/home/rzmws0/mhaller
  dos.fullname ('~mhaller//./../')
/home/rzmws0
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'

Returns
vector of strings

---

os.dirname - Function

Package	**utility**
Module	**misc**
Tool	**os**

get directory part in path names

Synopsis
dirname(pathname)

Description
After the path names are expanded to their full names (see above), it returns the directory part of each full name.

Example

```
  dos.dirname ('~mhaller')
/home/rzmws0
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'

Returns
vector of strings

---

os.basename - Function

Package	**utility**
Module	**misc**
Tool	**os**

get file part in path names

Synopsis
basename(pathname)

Description
After the path names are expanded to their full names (see above), it returns the file part of each full name.

Example

```
  dos.basename ('~mhaller')
mhaller
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'

Returns
vector of strings

---

os.filetype - Function

Package	**utility**
Module	**misc**
Tool	**os**

determine file type

Synopsis
filetype(filename, follow)

Description
Determine the file type of each file. It also recognizes tables.
By default it follows a symbolic link.
The string returned can have the following values:

Table

Directory

Regular File

SymLink

Invalid (meaning that the file does not exist)

Unknown

Note that in principle the type 'SymLink' can only be returned when `follow=F`.

Example

```
  dos.filetype ('~/aips++/test/hol.ms')
Table
```

Arguments

filename	in	file name Allowed:vector of strings
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of strings

---

os.filetime - Function

Package	**utility**
Module	**misc**
Tool	**os**

return the file time

Synopsis
filetime(pathname, whichtime, follow)

Description
Return for each file the time the file was last accessed (1), modified (2), or its status changed (3). Default is 2 (time modified).
The time returned is the Modified Julian Date (as UTC).
By default it follows a symbolic link.

Example

```
  dos.filetime ('~/aips++/test/hol.ms')
# 51638.4
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'
whichtime	in	which time to return Allowed:integer Default: 2
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of doubles or fail

---

os.filetimestring - Function

Package	**utility**
Module	**misc**
Tool	**os**

return the file time string

Synopsis
filetimestring(pathname, whichtime, follow)

Description
Return for each file the time the file was last accessed (1), modified (2), or its status changed (3). Default is 2 (time modified).
The time returned is the local time.
By default it follows a symbolic link.

Example

```
  dos.filetimestring ('~/aips++/test/hol.ms')
# 04-Apr-2000/12:43:27
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'
whichtime	in	which time to return Allowed:integer Default: 2
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of doubles or fail

---

os.size - Function

Package	**utility**
Module	**misc**
Tool	**os**

determine file or directory sizes

Synopsis
size(pathname, follow)

Description
Determine the file sizes in bytes. When a file is a directory, the size is the sum of the sizes of all files in the directory, including all files in subdirectories. It is similar to the UNIX command `du -s`.
By default it follows a symbolic link.

Example

```
  dos.size ('~/aips++/test/hol.ms')
# 2.59398e+07
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of doubles or fail

---

os.freespace - Function

Package	**utility**
Module	**misc**
Tool	**os**

determine freespace on file systems

Synopsis
freespace(pathname, follow)

Description
Determine the free space (in bytes) on the file systems containing the given files or directories.
By default it follows a symbolic link.

Example

```
  dos.freespace ('~/aips++/test/hol.ms')
# 1.09639e+09
```

Arguments

pathname	in	path name Allowed:vector of strings Default: '.'
follow	in	follow symlink? Allowed:bool Default: T

Returns
vector of doubles or fail

---

os.copy - Function

Package	**utility**
Module	**misc**
Tool	**os**

copy a file, directory, or symlink

Synopsis
copy(source, target, overwrite, follow)

Description
Copy a file, directory (recursively), or symbolic link.
If the source is a directory, the entire directory (including all subdirectories) is copied (similar to UNIX command `cp -r`). However, unlike that UNIX command the source directory itself is not copied. Only its contents are copied.
If the source is a file or symlink and the target is a directory, the source is copied to that directory (similar to UNIX command `cp`).
By default a source symbolic link is followed. In that case the symbolic link is not copied, but the file or directory it links to.
An error is given if the target already exists, unless `overwrite=T`. In that case the target will be removed first.

Example
This command copies the entire MeasurementSet to /aips++2/gvandiep. Note that the name of the MeasurementSet has to be given in the target.

```
dos.copy ('~/aips++/test/hol.ms', '/aips++2/gvandiep/hol.ms')
```

Arguments

source	in	path name of source Allowed:string
target	in	path name of target Allowed:string
overwrite	in	overwrite if target exists? Allowed:bool Default: F
follow	in	follow symlink? Allowed:bool Default: T

Returns
T or fail

---

os.move - Function

Package	**utility**
Module	**misc**
Tool	**os**

move a file, directory, or symlink

Synopsis
move(source, target, overwrite, follow)

Description
Move a file, directory (recursively), or symbolic link. It can also move across file systems.
If the source is a directory, the entire directory (including all subdirectories) is moved (similar to UNIX command `mv`). However, unlike that UNIX command the source directory itself is not moved. Only its contents are moved.
If the source is a file or symlink and the target is a directory, the source is moved to that directory (similar to UNIX command `mv`).

By default a source symbolic link is followed. In that case the symbolic link is not moved, but the file or directory it links to.
An error is given if the target already exists, unless `overwrite=T`. In that case the target will be removed first.

Example

This command moves the entire MeasurementSet to /aips++2/gvandiep. Note that the name of the MeasurementSet has to be given in the target.

```
dos.move ('/aips++2/gvandiep/hol.ms', '~/aips++/test/hol.ms')
```

Arguments

source	in	path name of source Allowed:string
target	in	path name of target Allowed:string
overwrite	in	overwrite if target exists? Allowed:bool Default: F
follow	in	follow symlink? Allowed:bool Default: T

Returns

T or fail

---

os.remove - Function

Package	**utility**
Module	**misc**
Tool	**os**

remove files, directories, or symlinks

Synopsis

remove(pathname, recursive, mustexist, follow)

Description

Remove one or more files, directories (optionally recursively), or symbolic links.

If the source is a directory, it will be by default be removed recursively. If `recursive=F` only the directory will be removed. In that case it results in an error if it is not empty.
By default a source symbolic link is followed. In that case not only the symbolic link is removed, but also the file or directory it links to.
By default an error is given if the target to be deleted does not exist. This can be prevented by giving `mustexist=F`.

Example
This command removes an entire MeasurementSet.

```
dos.remove ('~/aips++/test/hol.ms');
```

Arguments

pathname	in	path names Allowed:vector of strings
recursive	in	remove directories recursively? Allowed:bool Default: T
mustexist	in	does the file need to exist? Allowed:bool Default: T
follow	in	follow symlink? Allowed:bool Default: T

Returns
T or fail

---

os.lockinfo - Function

Package	**utility**
Module	**misc**
Tool	**os**

get info about table use and locking

Synopsis
lockinfo(tablename)

Description
Get info about use and locking of a table. If so, give the pid of the process using or locking it. The function `showtableuse` uses this function to print a message about the use and locking of a table.
It returns a vector of 3 integer elements. The first element tells if the table is used or locked.
0 = not used
1 = used, but not locked
2 = read-locked
3 = write-locked
The second element gives the pid of the process using or locking the table.
The third element is 1 if the lock is permanent (otherwise 0).
Note that multiple processes can use a table or hold a read-lock. This function gives the pid of only one of those processes.

Example
Note that this example is done better by function `showtableuse`.

```
res := dos.lockinfo ('~/aips++/test/hol.ms');
if (res[1] != 0) {
  print (paste ('The table is used (and maybe locked) by process', res[2]));
}
```

Arguments

tablename	in	table name Allowed:string

Returns
vector of 2 integers or fail

---

os.showtableuse - Function

Package	**utility**
Module	**misc**
Tool	**os**

tell if a table is used or locked

Synopsis
showtableuse(tablename)

Description
Tell if a table is used or locked in another process. A message is sent to the logger telling if the table is used or locked and by which process.
Note that multiple processes can use a table or hold a read-lock. This function gives the pid of only one of those processes.

Example

```
  dos.showtableuse ('~/aips++/test/hol.ms');
# NORMAL: Table ~/aips++/test/hol.ms is opened
# (but not locked) in process 5242
```

Arguments

tablename	in	table name Allowed:string

Returns
T or F

---

os.mail - Function

Package	**utility**
Module	**misc**
Tool	**os**

mail a message to a user

Synopsis
mail(message, recipient, sender, subject, cc, bcc)

Description
Mail a message to one or more recipients. It is also possible to give zero or more cc- and bcc-addresses. Note that `\n` can be used to create multiple lines in the message. If the sender is given as a boolean, it is assumed that the message contains lines containing the sender, subject, etc.. Otherwise the message is prepended with such lines.
The mailer to be used can be defined in the aipsrc variable `os.mailer`. It defaults to `mailx`.

The function is used by bug and ask. Currently it uses a shell command to do the mailing.

Example
The following command sends a message with noordam as the sender.

```
dos.mail ('line1\nline2', 'gvd mhaller', 'noordam', 'some subject',
          cc='tcornwel@nrao.edu wyoung@nrao.edu')
```

Arguments

message	in	message to send Allowed:string
recipient	in	whom to send the message Allowed:string
sender	in	whom the message comes from Allowed:string Default: T
subject	in	subject of the email Allowed:string Default: ''
cc	in	whom to cc the message Allowed:string Default: ''
bcc	in	whom to bcc (blind copy) the message Allowed:string Default: ''

Returns
T or fail

---

os.edit - Function

Package	**utility**
Module	**misc**
Tool	**os**

edit a file

Synopsis
edit(file, editor, async)

Description
This command starts a shell to edit a file. It can only be used in a GUI environment. An editor like vi should not be used, because it does not start up in a separate window. The shell may be synchronous or asynchronous (default). Use the synchronous option if the file is needed immediately.
The default editor is emacs.

Example

```
dos.edit ('file.cc');
```

Arguments

file	in	file to be edited Allowed:string
editor	in	editor to be used Allowed:string Default: emacs
async	in	Run asynchronously Allowed:boolean Default: T

Returns
T or fail

---

os.type - Function

Package	**utility**
Module	**misc**
Tool	**os**

Return the type of this tool

Synopsis
type()

Returns
string