File

COLLABORATORS			
	TITLE : File		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 31, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

File

1.1 File V1.00

```
Pure Basic - File library V1.00

  The file is the main way of storage for nowadays computers.
  With the PureBasic you manage them in very simple and optimized
  way. Any number of files can be handled at the same time. This
  library use buffered functions to increase the writing/reading
  speed.


Commands summary in alphabetical order:

   CloseFile
   CreateFile
   DeleteFile
   Eof
   FileSeek
   InitFile
   Loc
   Lof
   MakeDirectory
   OpenFile
   ReadByte
   ReadFile
   ReadLong
   ReadString
   ReadWord
   RenameFile
   UseFile
   WriteByte
   WriteLong
   WriteString
   WriteStringN
   WriteWord


Example:
```

```
    File demo
```

1.2 closefile

```
  Syntax

CloseFile(#File)

  Description

Close the specified #File and it can't be used anymore for later use.
Closing a file ensure the buffer will be put effictively on the disk.

Note: on the program end, Pure Basic is smart enough to close all
      the unclosed files, so you don't need to do it yourself.
```

1.3 createfile

```
  Syntax

Result = CreateFile(#File, FileName$)

  Description

Open an empty file. If the file was existing, it open it and replace it
by a blank one ! Careful. If 'Result' is not null, the file is effectively
created and you get the AmigaOS pointer to the file (for advanced programmers).
If Result is NULL, the creation has failed. It must be always tested, as
performing operations on a non created file will cause severe crashs.
```

1.4 deletefile

```
  Syntax

Result = DeleteFile(FileName$)

  Description

Delete the specified 'Filename$'. You can give the full path and filename
or only the filename (then the current directory will be used). If the
result is NULL, then the filename hasn't been deleted.
```

1.5 eof

```
  Syntax

Result = Eof()
```

Description

Eof stands for 'End Of File'. It will return true if you have reached the end of the current file, NULL else.

1.6 fileseek

Syntax

```
FileSeek(NewPosition)
```

Description

It designed to change the current pointer position inside the file.

1.7 initfile

Syntax

```
Result = InitFile(#NumMaxFiles)
```

Description

With init all the file envirronement for future use. You must call this function before any other file functions.

#NumMaxFiles = Number of maximum files you need to handle simultanely.

1.8 loc

Syntax

```
Position = Loc()
```

Description

Return the actual pointer position inside the file.

1.9 lof

Syntax

```
Length = Lof()
```

Description

Lof stands for 'Length Of File'. It will return the length of the current file.

1.10 makedirectory

Syntax

```
Result = MakeDirectory(Path$)
```

Description

Create the directory given in the Path$. If the result is NULL, then the directory can't be created.

1.11 openfile

Syntax

```
Result = OpenFile(#File, FileName$)
```

Description

It will open the designed file or create it if it doesn't exists. You can perform read and write on this file. If 'Result' is not null, the file is effectively opened and you get the AmigaOS pointer to the file (for advanced programmers). If Result is NULL, the creation has failed. It must be always tested, as performing operations on a non created file will cause severe crashs.

1.12 readbyte

Syntax

```
Number.b = ReadByte()
```

Description

Read one byte on the current opened file.

1.13 readfile

Syntax

```
Result = ReadFile(#File, FileName$)
```

Description

```
Open an existing file for read only operations. If 'Result' is not null, the ↩
    file
is effectively opened and you get the AmigaOS pointer to the file (for advanced ↩
    programmers).
If Result is NULL, the file is not found or can't be opened. It must be always ↩
    tested, as
performing operations on a non created file will cause severe crashs.
```

1.14 readlong

```
  Syntax

Number.l = ReadLong()

  Description

Read one long on the current opened file.
```

1.15 readstring

```
  Syntax

Text.s = ReadString()

  Description

Read one string on the current opened file.
```

1.16 readword

```
  Syntax

Number.w = ReadWord()

  Description

Read one word on the current opened file.
```

1.17 renamefile

```
  Syntax

RenameFile(OldFilename$, NewFilename$)

  Description

Rename a file. If the result is NULL, then the file can't be renamed.
```

1.18 usefile

Syntax

```
UseFile(#File)
```

Description

It change the current used file to the given one.

1.19 writebyte

Syntax

```
WriteByte(Number)
```

Description

Write a byte number inside the current file. File must be opened
with write feature (ie: not with ReadFile()).

1.20 writelong

Syntax

```
WriteLong(Number)
```

Description

Write a long number inside the current file. File must be opened
with write feature (ie: not with ReadFile()).

1.21 writestring

Syntax

```
WriteString(Text$)
```

Description

Write a string inside the current file. File must be opened
with write feature (ie: not with ReadFile()).

1.22 writestringn

Syntax

```
WriteStringN(Text$)
```

Description

Write a string inside the current file and add the 'end of line' character.
The file must be opened with write feature (ie: not with ReadFile()).

1.23 writeword

Syntax

```
WriteWord(Number)
```

Description

Write a word number inside the current file. File must be opened
with write feature (ie: not with ReadFile()).