parser_guide

COLLABORATORS			
	TITLE : parser_guide		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

parser_guide

1.1 AMIGA-E Module: parser_oo/MAIN

```
                                  ** PARSER_OO - Written By Fabio Rotondo **

                            ** DOCUMENTATION GUIDE **


                  Introduction

                 Author's Infos

                 Example Program
                    COMMANDS             BRIEF DESCRIPTION
    ------------------------------------------------------------------

                 parser(numels=25)
                 Inits the Parser Object.

                 error()
                 Returns the LAST error.

                 arg(number)
                 Gets the desired arg.

                 parse(temp, s)
                 Parses a string.


                 ERROR TABLE
```

1.2 AMIGA-E Module: parser_oo/parser()

```
                         NAME: parser(numelements=25)

 DESCRIPTION: This command initializes a Parser Object.
```

```
      INPUTS: numelements - (optional) This param will allow you to define
                            maximum number of elements to be stored.

     RESULTS: NONE.

    SEE ALSO:
                 error()
```

1.3 AMIGA-E Module: parser_oo/error()

```
                         NAME: error()

 DESCRIPTION: This command returns the LAST error occurred.

      INPUTS: NONE.

     RESULTS: Error value.

    SEE ALSO:
                 ERROR TABLE
```

1.4 AMIGA-E Module: parser_oo/arg()

```
                         NAME: arg(argnum)

 DESCRIPTION: Returns the parsed argnum.

      INPUTS: argnum - Number of arg to obtain.

     RESULTS: a PTR TO LONG conatining the STRING/LONG desired.

    SEE ALSO:
                 parse()
```

1.5 AMIGA-E Module: parser_oo/parse()

```
                         NAME: parse(temp:PTR TO CHAR, string:PTR TO CHAR)

 DESCRIPTION: This command parses the passed 'string' using the template
              provided by 'temp'.

      INPUTS: temp    - PTR TO CHAR containing the Template to use in parsing.

              string  - PTR TO CHAR conatining the string to parse.

     RESULTS: TRUE    - Correct Parsing.
              FALSE   - Error in Parsing.
```

```
    SEE ALSO:
                error()

                arg()
```

1.6 AMIGA-E Module: parser_oo/ERROR TABLE

```
VALUE       NAME            DESCRIPTION
-------------------------------------------------------------------------
0           NO_ERR          No Error.
1           ALLOC_ERROR     Failed to AllocDosObject().
2           NO_ARGS         There is no DosObject
```

1.7 Author's Infos

```
    My name is Fabio Rotondo. I am a free-lance Amiga programmer and I would
    like to get in touch with anyone who writes code for the Amiga.
    I write in AmigaE, BlitzII, C and a bunch of other languages.

    Please, feel free to contact me for any suggestions/questions.

    My address is:

    Fabio Rotondo
    C.so Vercelli 9
    28100 Novara
    ITALY
    Tel. (ITA) - (0)321 - 459676
    e-mail: fabio.rotondo@intercom.it

    Thanks!
```

1.8 Example Program

```
MODULE 'Fabio/Parser_oo'  -> This is our MAGIC module

PROC main()
  DEF par:PTR TO parser   -> Whoa! A Parser Object!!

  NEW par.parser()        -> Let's Initialize It

  IF (par)                -> If it exists, let's try a little parsing...
    IF (par.parse('SOURCE/A,DEST/A,COLOR/K','Ram:Pippo COLOR Blue Work:'))
      WriteF('Source:"\s"\n',par.arg(0))  -> Get the Arg #0
      WriteF('  Dest:"\s"\n',par.arg(1))  -> Get the Arg #1
      WriteF(' Color:"\s"\n',par.arg(2))  -> Get the Arg #2
    ELSE
      WriteF('Parsing Error: \d\n', par.error())  -> Ouch!!! An Error!
    ENDIF
```

```
  ELSE
    WriteF('Init Error\n')                          -> Ouch!!! Another Error!!
  ENDIF

  END par       -> Remember ALWAYS to END this Object BEFORE exiting!!!
  CleanUp(0)    -> Let's keep things clean...
ENDPROC
```

1.9 AMIGA-E Module:parser_oo/INTRODUCTION'

```
                    PARSER_OO is a 'strange' class. It uses the ReadArgs function  ↩
                       to parser
    standard string with a given template. In other words, you can use the
    power of dos/ReadArgs() to create smart string parser control without
    ANY line of code!!!

    All you have to do is simply call the command parse() with a Template
    and the string you want to parse... and this object will do the trick!

    To know how to write a Template, have a nice look to AutoDocs
    Dos/ReadArgs() function. And always remeber the args counting starts from
    0 not 1!!!

    So, for example:

    TEMPLATE: 'SOURCE/A,DEST/A,MODE/A'

               SOURCE is arg(0), DEST is arg(1), MODE is arg(2)


    Read the
                 example program
                .

    If you like this module and use it, please consider contact me
    at the address you'll find in
                 Author's Infos
```