NodeMaster

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

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

NodeMaster

1.1 Amiga-E Module: NodeMaster

```
                              ** NodeMaster_OO - Written By Fabio Rotondo **

                           ** DOCUMENTATION GUIDE **


                  Introduction

                  Author's Infos

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

                  nodemaster()
                   Initializes the NodeMaster object

                  add(object, mode=SN_ADD_TAIL)
                   Add obejct to the list

                  addr()
                   Get the pointer to the Exec list

                  change(object)
                   Change current node object

                  changepos(node)
                  Change current node position

                  clear()
                  Clear ALL list

                  del()
                   Delete the current node

                  empty()
                  Check if there are nodes in list

                  first()
```

```
 Jump to the first node

get()
 Get the pointer to the current node

insert(string)
 Insert an object AFTER the current one

item(numitem)
 Jump to the specific item

last()
 Jump to the last node

obj()
 Get the PTR of the current object

numitems()
 Return number of nodes in memory

pop()
 Get a node from the stack

prev()
 Go to the previous node

push()
 Push a node on the stack

succ()
 Go to the next node
```

1.2 Introduction

NodeMaster is a generic object handling module which will allow you to easily create lists of whatever you want. Its generic structure is ideal for polimorphism and inheritance in other objects. This object just take care of creating Exec Lists and Nodes of something you pass to it (we will call it "object") and then it has great power in Lists manipulation. You can easily add/del nodes from the list, go to a specific item by its ordinal number and so on...

I have worked out this module structuring it on Amiga Exec's Nodes. This means that everything you will add/remove to a NodeMaster will be done on a System List Node.

This code is very Exec List-based, so you can do whatever you want.

Main features are:

* Push/Pop commands to save/restore a special node position.

* Insert command to add a object AFTER another (usually it is added as Last node)

* Search command to scan trought the list looking for a string.

* Item   command to go to a specific item by its ordinal number.

1.3 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: fsoft@intercom.it

Check out my WWWPage with many of my AmigaE Modules/Sources!

 http://www.intercom.it/homepages/utenti/fsoft/index.html

Thanks!

1.4 Example Program

SORRY, due to the flexible state of this object, no source is provided.

PLEASE, refer to StringNode.lha archive for a demo of NodeMaster and
StringNode objects (this one inherits all methods of NodeMaster ;)

Thank You!

1.5 AmigaE Modules: NodeMaster/NodeMaster()

       NAME: NodeMaster()

DESCRIPTION: Use this command to inizialize a NodeMaster object.

      INPUT: NONE.

    RESULTS: NONE.

   SEE ALSO:

1.6 AMIGA-E Modules: NodeMaster/add()

```
                        NAME: add(object:PTR TO LONG, mode=SN_ADD_TAIL)

 DESCRIPTION: Use this command to add an object to the list.

       INPUT: object         - PTR TO LONG. This is the object to add.
              mode           - (OPTIONAL) This flag is very useful to
                               choose _where_ a new node will be added.
                               Default is as last one, but you can
                               add it as the first line or in the middle
                               of the list (same as insert() command).

                               Possible values are:

                               SN_ADD_HEAD - Use this one to add the node
                                             as the first in list.

                               SN_ADD_HERE - Use this one to add the node
                                             AFTER the current one.
                                             (Same as insert() method)

                               SN_ADD_TAIL - (Default) Use this one to add
                                             the node as the last in list.

         NOTE: It is not safe to use mode SN_ADD_HERE (same as insert) if you
               do not know what you are doing: if there are no items
               unpredictable results may occur... it could also work
               properly, but I have not tested! ;)

      RESULTS: TRUE       - The node has been added correctly.
               FALSE      - Something went wrong (usually memory problems)

    SEE ALSO:
               insert()

               del()
                                  name()

               item()
```

1.7 Amiga-E Modules: NodeMaster/addr()

```
                        NAME: addr()

 DESCRIPTION: Use this command to get the addr of the Exec List.

       INPUT: NONE.

      RESULTS: A PTR TO lh  (an Exec List Header).

         NOTE: This command is useful expecially with ListView gadgets
               which requires a PTR TO an Exec List Header. All you have
               to do is:
```

```
                Gt_SetGadgetAttrsA(listgad, win, req, [
                    GTLV_LABELS, NodeMasterobj.addr(), 0,0])

            And the ListView will show your new list.
            Please refer to RMKM and autodocs for more infos regarding
            Gadtools and ListViews.

   SEE ALSO:
            get()
```

1.8 Amiga-E Modules: NodeMaster/first()

```
                        NAME: first()

DESCRIPTION: Use this command to jump to the first object in the list.

      INPUT: NONE.

    RESULTS: TRUE           - Position correct.
             FALSE          - Cannot go to the first (maybe list empty).

   SEE ALSO:
             last()
                              name()

             del()
                              search()
```

1.9 Amiga-E Modules: NodeMaster/del()

```
                        NAME: del()

DESCRIPTION: Use this command to delete the current node.
             After deletion the CURRENT NODE will be the next one.
             If the node you deleted was the last one, then the next
             will be the previous one.

      INPUT: NONE.

    RESULTS: NONE.

   SEE ALSO:
             clear()

             add()
```

1.10 Amiga-E Modules: NodeMaster/get()

```
                        NAME: get()

DESCRIPTION: Use this command to get a pointer to the current Exec List
             node.

      INPUT: NONE.

    RESULTS: A PTR TO ln (An Exec List Node)

   SEE ALSO:
             addr()
```

1.11 Amiga-E Modules: NodeMaster/insert()

```
                        NAME: insert(object:PTR TO LONG)

DESCRIPTION: Use this command to add an object AFTER the current node.

      INPUT: object     - PTR TO CHAR. Object you want to add.

    RESULTS: NONE.

       NOTE: The current node WILL NOT change!!!

   SEE ALSO:
             add()
                                name()

             del()

             item()
```

1.12 Amiga-E Modules: NodeMaster/item()

```
                        NAME: item(numitem)

DESCRIPTION: Use this command to position current node to the ordinal
             numitem node.

      INPUT: numitem     - LONG. Ordinal value of node position.

    RESULTS: NONE.

   SEE ALSO:
             numitems()
```

1.13 Amiga-E Modules: NodeMaster/last()

```
                        NAME: last()

DESCRIPTION: Use this command to position current node to the last one.

      INPUT: NONE.

    RESULTS: NONE.

   SEE ALSO:
             first()

             item()
```

1.14 Amiga-E Modules: NodeMaster/obj()

```
                        NAME: obj()

DESCRIPTION: Use this command to get the current node's PTR TO object.

      INPUT: NONE.

    RESULTS: PTR TO LONG to the current node object.

   SEE ALSO:
             add()
```

1.15 Amiga-E Modules: NodeMaster/numitems

```
                        NAME: numitems()

DESCRIPTION: Use this command to know how many items are added to the
             list.

      INPUT: NONE.

    RESULTS: items  - LONG. Number of items.

   SEE ALSO:
             add()

             del()
```

1.16 Amiga-E Modules: NodeMaster/pop()

```
                        NAME: pop()

DESCRIPTION: Use this command to restore current node to the one
             previously Push()ed.
```

```
   INPUT: NONE.

 RESULTS: NONE.

    NOTE: * If no node was Push()ed the current node won't change.

SEE ALSO:
          push()
```

1.17 Amiga-E Modules: NodeMaster/prev()

```
                    NAME: prev()

DESCRIPTION: Use this command to go to the previous string in the list.

   INPUT: NONE.

 RESULTS: TRUE   - Positioning successful.
          FALSE  - No previous items.

SEE ALSO:
          succ()

          first()

          last()
```

1.18 Amiga-E Modules: NodeMaster/push()

```
                    NAME: push()

DESCRIPTION: Use this command to memorize the current node position.

   INPUT: NONE.

 RESULTS: NONE.

    NOTE: * Push()ing a position after another will cause the lost of
            the previous Push()ed position.
            (Stacked pushed are not supported (yet ;))


SEE ALSO:
          pop()

          first()

          last()
```

1.19 Amiga-E Modules: NodeMaster/succ()

```
                        NAME: succ()

DESCRIPTION: Use this command to position current node to the next one in
             list.

      INPUT: NONE.

    RESULTS: TRUE       - Positioning successful.
             FALSE      - No next items.

   SEE ALSO:
             prev()
```

1.20 Amiga-E Modules: NodeMaster/change()

```
        NAME: change(object:PTR TO LONG)

DESCRIPTION: Use this command to change current node PTR TO object.

      INPUT: object  - PTR TO LONG. New object to change with the old one.

    RESULTS: NONE.

   SEE ALSO:
```

1.21 Amiga-E Modules: NodeMaster/clear()

```
                        NAME: clear()

DESCRIPTION: Use this command to clear all items in the list.

      INPUT: NONE.

    RESULTS: The list will be completely empty.

   SEE ALSO:
             del()
```

1.22 Amiga-E Modules: NodeMaster/empty()

```
        NAME: empty()

DESCRIPTION: Use this command to check whether the list is empty or not.

      INPUT: NONE.

    RESULTS: TRUE       - List is empty
             FALSE      - At least one item is present.

   SEE ALSO:
```

1.23 Amiga-E Modules: NodeMaster/changepos()

```
                        NAME: changepos(node:PTR TO ln)

DESCRIPTION: Use this command to change current node position to another.

      INPUT: node   - (PTR TO ln) new list node to change position to.

       NOTE: You *MUST* know exactly what you are doing. Passing a wrong
             node as parameter could get to Software Failures and so on.
             This command is designed only for "professional" user who
             intend build new object inheriting this one.

    RESULTS: The current node position will be changed.

  SEE ALSO:
             first()

             last()

             item()
```