

Executive

Petri Nordlund

Copyright © CopyrightÂ©1995 Petri Nordlund. All rights reserved.

COLLABORATORS

	<i>TITLE :</i> Executive		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Petri Nordlund	August 24, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Executive	1
1.1	Main menu	1
1.2	Copyright	2
1.3	Disclaimer	4
1.4	Acknowledgments	5
1.5	Betatesters and catalog translators	5
1.6	Introduction	6
1.7	Getting started	7
1.8	Multitasking basics	8
1.9	Different kinds of tasks	9
1.10	Scheduling	9
1.11	Executive features and requirements	12
1.12	How does it work?	14
1.13	Patched functions	15
1.14	Installed files	16
1.15	Common problems	17
1.16	Background	20
1.17	Server	20
1.18	Starting the server	21
1.19	Making it work	21
1.20	Server options	24
1.21	Quitting the server	26
1.22	Preferences file	27
1.23	Server Internals	29
1.24	Clients	33
1.25	Schedulers	35
1.26	Accounting	38
1.27	Genies	39
1.28	Glossary	40
1.29	Registration	42

1.30	Keyfile	45
1.31	Author	45
1.32	PGP KEY	46
1.33	History	47
1.34	Acct	53
1.35	Acct options	54
1.36	ALoad	56
1.37	ALoad options	57
1.38	ALoad3D	60
1.39	ALoad3D options	61
1.40	Commander	63
1.41	Commander options	65
1.42	Ctp	66
1.43	Ctp options	67
1.44	ExecutivePrefs	68
1.45	ExecutivePrefs options	71
1.46	Kill	72
1.47	Kill options	73
1.48	Lastcomm	75
1.49	Lastcomm options	76
1.50	Meter	77
1.51	Meter options	80
1.52	Nice	84
1.53	Nice options	85
1.54	Ps	86
1.55	Ps options	88
1.56	Ps Keywords	93
1.57	Pstree	100
1.58	Pstree options	101
1.59	Renice	102
1.60	Renice options	104
1.61	Sa	105
1.62	Sa options	107
1.63	Timer	110
1.64	Timer options	112
1.65	Top	113
1.66	Top options	116
1.67	Stat	119
1.68	Stat options	121
1.69	Uptime	121
1.70	Uptime options	122
1.71	About custom shells	124

Chapter 1

Executive

1.1 Main menu

E x e c u t i v e

Copyright

Disclaimer

Acknowledgments

Introduction

Common problems

Background

Server

Clients

Schedulers

Accounting

Genies

Glossary

Registration

Author

History

```
Acct
ExecutivePrefs
Ps
Stat
ALoad
Kill
Pstree
Timer
ALoad3D
Lastcomm
Renice
Top
Commander
Meter
Sa
Uptime
Ctp
Nice
by Petri Nordlund
```

1.2 Copyright

COPYRIGHT

~~~~~

Copyright © 1995 Petri Nordlund.

Executive software and logo are Copyright © 1995 Petri Nordlund.  
All rights reserved.

LICENCE

~~~~~

Executive is not, nor has ever been, public domain or free software.

Users are granted to use the unregistered version of Executive in order
to determine if it suits their needs.

You are granted the right to share Executive with others, as long as you distribute the Executive archive exactly as you received it, with all associated files included. REGISTERED USERS MAY NOT DISTRIBUTE THE SEPARATE FILE "Executive.key".

Under no circumstances may you charge more than a minimal copying fee or receive any other form of consideration for distributing the Executive files without express written consent from the copyright holder.

Bulletin Board system operators may post the unregistered Executive software on their BBS for downloading by their users without written permission only if the above conditions are met, and only if no special fee is necessary to access the Executive files. A general fee to access the BBS is ok.

Reverse engineering of the Executive software protection is strictly forbidden.

Several digital fingerprints are included in the "Executive.key" file and redistributed copies are easy to identify. Any redistribution of this file will lead to legal actions.

Distributing beta-versions of Executive is strictly prohibited.

Executive logo can be used for advertising purposes.

GADLAYOUT

~~~~~

Commander, ExecutivePrefs and Timer make use of the GadLayout dynamic gadget layout system by Timothy Aston.

#### MUI

~~~~~

This application uses

MUI - MagicUserInterface

(c) Copyright 1993/94 by Stefan Stuntz

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US\$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY

MAGICWB
~~~~~

The MagicWB is Copyright © 1992-94 Martin Huttenloher.

TRADEMARKS  
~~~~~

Amiga and Workbench are trademarks of Amiga Technologies GmbH.

VAX/VMS is a trademark of Digital Equipment Corporation.

UNIX is a trademark of X/Open Company Ltd.

OS/2 is a trademark of International Business Machines Corporation.

SAS/C is a trademark of SAS Institute Inc.

Postscript is a trademark of Adobe Systems Inc.

Final Writer is a trademark of SoftWood Inc.

Macintosh is a trademark of Apple Computer, Inc.

1.3 Disclaimer

DISCLAIMER
~~~~~

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

## 1.4 Acknowledgments

### ACKNOWLEDGMENTS

~~~~~  
First I must thank all

betatesters and catalog translators

.

Also, I want to thank the following people (in alphabetical order):

Osma Ahvenlampi

for ideas (SysInfo.library).

Timothy J. Aston

for GadLayout.

Pascal Eeftinck

for many ideas and comments.

Nico François

whose Cat2H utility solved all the problems I had with catalogs.

HiSoft

I have used their Devpac assembler for many years now and have been fully satisfied. Other assemblers still come far behind.

Wolfgang Loske

for numerous ideas and comments.

Richard Stallman & Roland McGrath

and others for GNU Make.

Stefan Stuntz

for MagicUserInterface.

Teemu Suikki

for ideas and comments.

Geert Uytterhoeven

for MultiUser.

Markus Wild, Philippe Brand, Lars Hecking, Fred Fish & Leonard Norrgard

and others for porting GCC to AmigaDos.

Thanks to everybody who has reported bugs and given suggestions and

ideas. These are always welcome.

1.5 Betatesters and catalog translators

BETATESTERS AND CATALOG TRANSLATORS

~~~~~

---

## BETATESTERS

|                   |                                           |
|-------------------|-------------------------------------------|
| Lauri Aalto       | <aalto@lyseo.otol.fi>                     |
| Osma Ahvenlampi   | <Osma.Ahvenlampi@hut.fi>                  |
| Doug Dyer         | <dyer@alx.sticommet.com>                  |
| Pascal Eeftinck   | <arcade@mystical.xs4all.nl>               |
| Georges Goncalves | <georges.goncalves@ramses.fdn.org>        |
| Mike Hellers      | <hellers_m@istvax.ist.lu>                 |
| Andreas Jung      | <ajung@hssun5.cs.uni-sb.de>               |
| Mathias Karlsson  | <d3karma@dtek.chalmers.se>                |
| Wolfgang Loske    | <wloske@cip.e-technik.uni-erlangen.de>    |
| Koen Peetermans   | <Koen.Peetermans@ping.be>                 |
| Marcin Orlowski   | <carlos@felix.univ.szczecin.pl>           |
| Antonio Santos    | <L38058@alfa.ist.utl.pt>                  |
| Christoph Stoppe  | <stoppe@buche.informatik.uni-dortmund.de> |
| Teemu Suikki      | <zuikkis@zuikkis.pp.sci.fi>               |

## CATALOG TRANSLATORS

|                   |                                           |            |
|-------------------|-------------------------------------------|------------|
| Pascal Eeftinck   | <arcade@mystical.xs4all.nl>               | Dutch      |
| Georges Goncalves | <georges.goncalves@ramses.fdn.org>        | French     |
| Mathias Karlsson  | <d3karma@dtek.chalmers.se>                | Swedish    |
| Petri Nordlund    | <petrin@megabaud.fi>                      | Finnish    |
| Marcin Orlowski   | <carlos@felix.univ.szczecin.pl>           | Polish     |
| Antonio Santos    | <L38058@alfa.ist.utl.pt>                  | Portuguese |
| Christoph Stoppe  | <stoppe@buche.informatik.uni-dortmund.de> | German     |

## 1.6 Introduction

### INTRODUCTION

~~~~~

Getting started

Multitasking basics

Different kinds of tasks

Scheduling

Executive features and requirements

How does it work?

Installed files

If you want to be informed when a new version of Executive is released,

please send your email address to me

.

1.7 Getting started

GETTING STARTED

~~~~~

If you have installed Executive normally and rebooted your system, it's now running and you can proceed. If not, go ahead and install it, it's very easy to uninstall using the uninstall script.

Executive consists of a  
server  
and  
clients  
. The server is run in the  
background and most of the clients require it to be running to work.

In a few words, Executive adds a better task scheduler to your Amiga and provides similar services as programs like Job Manager or SPY system.

No similar programs for Amiga have been made before, so you may not exactly know what Executive is all about, but give it a little time and read the documentation and you won't regret it!

### UNREGISTERED VERSION

These features require a registered version of Executive:

- \* All schedulers except STANDARD
- \* Accounting
- \* ALoad3D
- \* Meter

The absence of these features doesn't make Executive unusable. You can use Executive without these features but I hope you'd  
register  
after  
you have seen what this program is.

### BUGS

If you find a bug or have a suggestion for improving Executive please contact me.

If you want to report a bug, please include the following information:

- Your Amiga hardware (Amiga type, CPU, memory)
- Your AmigaDos operation system version
- Other relevant software running on your system
- Executive version

If you can find the necessary steps to reproduce the bug, that would help me to locate the bug.

---

## 1.8 Multitasking basics

### MULTITASKING BASICS

~~~~~

Multitasking is something that Amiga users have been enjoying for 10 years while others are still arguing if multitasking is a useful feature in an operating system. You are probably familiar with multitasking, but I'd like to clarify some terms and draw an overall picture of different ways to implement multitasking. This will help you understand what Executive has to offer for you.

MULTITASKING ON AMIGA

The kind of multitasking Amiga has is called preemptive prioritized Round-Robin. Each task has a priority assigned to it. The task that has the highest priority and is ready to run, will be run. A task can be in one of three states: sleeping, ready or running.

- * A ready task is not currently using the CPU but is waiting to use it. Exec keeps a sorted list of the tasks that are ready. The list is sorted according to task priority, so Exec can easily find the ready task with the highest priority.
- * A sleeping task is not currently running and is waiting for some event to happen. When that even occurs, Exec will move the sleeping task into the list of ready tasks.
- * A running task is currently using the CPU. It will remain the current task until one of these three things occur:
 - o A higher priority task becomes ready, so Exec preempts the current task and switches to the higher priority task.
 - o The currently running task needs to wait for an event, so it goes to sleep and Exec switches to the highest priority task in Exec's ready list.
 - o The currently running task has had control of the CPU for at least a preset time period called a quantum and there is another task of equal priority ready to run. In this case, Exec will preempt the current task for the ready one with the same priority. This is known as time-slicing or "Round-Robin". When there is a group of tasks of equal priority on the top of the ready list, Exec will cycle through them, letting each one use the CPU for a quantum.

The term preemptive prioritized Round-Robin comes from these three features.

PREEMPTIVE VS. COOPERATIVE

Preemptive multitasking means that the operating system can force a task to give the CPU to another task. Some operating systems have cooperative multitasking, mainly because they are old and their developers didn't see

any need for better multitasking (or for any kind of multitasking) when the OS was developed. It's usually not possible to implement preemptive multitasking afterwards, so cooperative multitasking must be used.

In cooperative multitasking operating system tasks must handle the multitasking, i.e. do someone else's homework.

PRIORITIES

Amiga has fixed priorities. Most user tasks have a priority -1, 0 or 1. Tasks that require CPU time before user tasks, for example device drivers, have a priority 5. Harddisk partitions and disk drives have a priority 10 and the highest priority is usually 20, for input.device that handles all input from mouse and keyboard.

The priority range is -128 to +127. Negative priorities are usually used for tasks that require all the CPU time they can get, for example rendering programs. As a general rule, you shouldn't raise the priority of any normal task above 3.

1.9 Different kinds of tasks

DIFFERENT KINDS OF TASKS

Tasks can be divided in to two categories:

- * transput-bound
- * compute-bound

Transput-bound tasks spend most of their time waiting for some event to occur. Transput-bound tasks can be interactive or non-interactive. An example of an interactive transput-bound task is a text editor, it spends most of the time sleeping, waiting for a keypress. Device drivers are non-interactive transput-bound tasks.

Compute-bound tasks spend their time using as much CPU time as they can get. Compute-bound task can also be interactive. An example of this kind of task is a program that lets you rotate a three dimensional object on the screen with the mouse. Most compute-bound tasks are non-interactive, like a rendering program or a compiler.

1.10 Scheduling

SCHEDULING

The goal of scheduling is to provide good service to all tasks that are currently competing for the computing resource, that is, the execution of instructions.

DIFFERENT TASKS HAVE DIFFERENT NEEDS

A personal computer like Amiga usually has non-interactive transput-bound tasks, interactive transput-bound tasks and non-interactive compute-bound tasks. Different kinds of tasks must be given different treatment:

- * consistent response time for interactive transput-bound tasks
- * small response time for non-interactive transput-bound tasks
- * good throughput for compute-bound tasks

Interactive transput-bound tasks require immediate response to every event, otherwise the user might get very frustrated. Failing that, consistent response is better than good average response. Response to a keypress that always takes about 2 seconds is better than response that averages about 1 second but occasionally takes 10 or only 0.5.

Non-interactive transput-bound tasks require fast response to events, because otherwise they'll be delayed. For example, consider a task that needs to compute for 1 millisecond and then wait 20 milliseconds for an event to occur. This will be repeated 1000 times. This would require 1 second of computation and 20 seconds of waiting, a total of 21 seconds. But if the task is delayed half a second every time it becomes ready, it will take 521 seconds. A small average delay will pay off handsomely, even if some delays are longer. A 1-millisecond average delay will allow the task to finish in 22 seconds.

Compute-bound tasks require lots of CPU time, and the overhead should be minimized, i.e. the number of task switches should be minimized. A context switch on AmigaDos is much faster than in some larger operating systems. Here are some context switch speeds for different machines/operating systems (in microseconds):

26	25MHz Amiga A4000/040 AmigaDos 3.1
106	66MHz Snake HP-UX 9.x
128	40MHz Viking SunOS 4.1.3
150	25MHz Amiga A2000/030 AmigaDos 2.1
210	33MHz 486 386BSD 0.1
212	50MHz RIOS AIX 3.2

HOW DO WE DISTINGUISH DIFFERENT KINDS OF TASKS?

This is a bit technical and you may want to
skip
it.

To make scheduling decisions, the scheduler must first distinguish the category where a certain task belongs. This can be very difficult because a task can drift from one category to another while it's running.

For example, a text editor is usually interactive transput-bound task, but when user starts a search & replace-function, it might become interactive compute-bound task. When it's saving text to a disk, it becomes non-interactive transput-bound. This means that the scheduler must adapt to different situations while the task is running. The best scheduler would be the one that could predict the future.

When implementing a scheduler the usual approach is to give priorities to each task and then modify these while the task is running. The simplest form of this is a decreasing priority scheduler where the priority of a task decreases continuously as it uses processor time, and the scheduler runs higher priority tasks first. Operating system called Multics used such a scheduler and discovered its major disadvantage, that on a heavily loaded system with significant numbers of short-lived tasks the priority of a lengthy task can decrease to a point where little or no further processor time is available to it.

To avoid the problem of permanently depressed priorities it is necessary to elevate them in some manner. There are two major choices for elevation methodologies: event-based elevation and processor usage aging. Event-based elevation is used to deliberately favor interactive response over compute-bound tasks because task priority is elevated by the occurrence of events such as keypress. This methodology assumes that tasks are either distinctly compute-bound or distinctly interactive (transput-bound) and that interactive tasks are more important. Tasks whose interactive work consumes large amounts of processor time may not do well under this methodology. The VAX/VMS scheduler employs this elevation methodology.

The second priority elevation methodology is processor usage aging. A scheduler that uses this methodology elevates priorities by gradually forgetting about past processor usage, usually in an exponential fashion. As a result, the scheduler's measure of processor usage is an exponentially weighted average over the lifetime of a task. A simple exponential average is not desirable because it has the unexpected side effect of raising priorities when the load on the system rises. This happens because under higher load each task gets a smaller share of the processor, so its usage average drops, causing its priority to rise. These elevated priorities can degrade system response under heavy loads because no task accumulates enough usage to drop its priority. The 4.3BSD version of UNIX solves this problem by making the aging rate depend on the load average, so that aging is slower in the presence of higher load, keeping priorities in approximately the same range.

DOES AMIGA NEED A BETTER SCHEDULER?

A better scheduler would be useful. It would free you from constantly adjusting task priorities.

There are many situations where Amiga slows down so much that it basically becomes a single-task system. For example, you might be printing a large document from a wordprocessor and it would take too much time to load another program so you'll just wait for the wordprocessor to finish.

When you start a time consuming task, you have to manually lower its priority before or after so that you can run other programs. For a 'power user' this soon becomes a nuisance.

It's also easy to lock up the system by raising the priority of a task, that wants to use all processor time, too high.

I have also run into problems with text editors. Their priority is usually 1 or 2, so they are above normal CLI-tasks. When you load

a large text-file to a text-editor and start a time consuming operation, for example macro execution or a vertical cut in CygnusEd, you can't do anything else before the operation finishes.

A better scheduler would handle situations like these automatically, without any user interaction.

1.11 Executive features and requirements

EXECUTIVE FEATURES AND REQUIREMENTS

Executive is

- * a scheduler
- * a task management system

Executive is similar to Job Manager or Spy System, but it has more features than these two together!

Executive doesn't use any illegal programming, it is fully compatible with all Amiga computers running AmigaDos release 2.04 or newer.

Executive has been carefully designed to work with all kinds of configurations and care has been taken not to break any programming rules.

Executive is a client-server application. The server is usually started when the system is booted. It will install the scheduler (optional) and collect statistics of CPU usage. Clients are used to query information from the server.

Here's a list of major features in Executive:

- * 6 different schedulers
- * focus - task with active window gets more CPU time
- * accounting
- * load averages
- * accurate (1/1000 seconds) CPU usage timing
- * tracks child-parent task relationships
- * process identifiers, process groups
- * support for
 - multiuser.library
 - * SysInfo.library for user-implemented clients

Other important things:

- * localized
- * on-line help
- * Most GUI-based clients available with GadTools and MUI interfaces
- * server and clients use memory pools, no memory fragmentation
- * works with Nicola Salmoria's SetManager

Here's a list of client-programs included:

Acct	Accounting daemon
ALoad	Display load average or CPU usage
ALoad3D	Display load averages in 3D
Commander	General task manager
Ctp	Extended ChangeTaskPri
Kill	Extended Break
Lastcomm	List last commands executed
Meter	Display system information
Nice	Run programs with lower scheduling priority
Ps	Process status displayer
Pstree	Display child-parent relationships
Renice	Renice a task
Sa	Display accounting statistics
Stat	Display some misc. information
Timer	Time CLI and Workbench programs
Top	Display information about the top CPU tasks
Uptime	Display system uptime and load averages

Will those clients run from CLI only?

A student asked the master for help... does this program run from the Workbench? The master grabbed the mouse and pointed to an icon. "What is

this?" he asked. The student replied "That's the mouse". The master pressed control-Amiga-Amiga and hit the student on the head with the Amiga ROM Kernel Manual.

-- Amiga Zen Master Peter da Silva

1.12 How does it work?

HOW DOES IT WORK?

~~~~~

The first thing that must be done is to start the server. Most of the clients require that the server is running because they need to get information from it. Server should be started as early as possible, preferably in startup-sequence, but it can be started and stopped later too.

When the server starts, it reads the preferences file. Most of the options can also be specified in tooltypes and on command line. You can modify the preferences file with the  
 ExecutivePrefs  
 preferences program.

The server will patch  
 some functions  
 in exec.library and intuition.library.

It necessary to run the server at a very high priority, currently at 126, otherwise the CPU usage calculations would not be accurate. This won't cause any problems with other tasks because Executive needs the CPU only for a very brief moment.

Executive doesn't replace Exec task dispatching functions, it just recalculates task priorities for scheduled tasks every second. Two priority ranges are needed. The first one is called "catch range" and the other is "dynamic range".

Catch range

Tasks that have priority inside the catch range will be automatically scheduled. The default range is from -105 to +2.

Dynamic range

When the scheduler calculates a new priority for a task, it will be in this range. The default range is from -100 to -50. This means, that a task that uses little or no CPU time, will get priority close to -50. Tasks that use much CPU time will have priority near -100.

So, not all tasks are scheduled automatically. You can still run programs with priority above and below the catch range. For example, input.device will run at priority 20 as usual.

Different schedulers calculate the priority in different way, but they all use a nice-value associated with each task. Nice-value is usually 0, but it can be used to give more CPU time to some tasks. The nice-value

is in range -20 to +20. Unlike priorities on Amiga, a NEGATIVE value gives MORE CPU time to a task, i.e. it will be less nice to other tasks.

Another thing common to all schedulers is the focus-feature. The task that has opened the currently active window gets more CPU time than others. This feature can be turned off.

When using a scheduler you can specify certain tasks that will be watched and some action will be taken when a task with the name you specified is created. This feature is needed when you don't want to schedule a task whose priority is in the catch range or you especially want to schedule some task that has priority outside the catch range. It's also possible to give default nice values or priorities for tasks when they are created.

The main use for this is to make sure that some programs like telecommunication programs are not scheduled, because that might drop their priority too low and cause errors in serial transfer.

The watched tasks feature is also used to specify certain tasks that should totally be ignored. There's currently only one task that must be ignored. The SAS/C Development System has a debugger called CPR which creates a task called 'VISOR input processor'. This task is not removed in a legal way (RemTask() is not called), so it will cause problems with programs like Executive. But this will be handled AUTOMATICALLY by Executive, so there's no need to worry about it.

There's absolute no need to ignore any other tasks.

When the server is running, you can start up some clients, for example Top and ALoad. While a client is running, you can't quit the server.

## 1.13 Patched functions

### PATCHED FUNCTIONS

~~~~~

exec.library

AddTask()
RemTask()
SetTaskPri()
Switch()

intuition.library

CloseWindow() - when FOCUS feature is ON
OpenWindow() - when FOCUS feature or OpenWorkbench fix is ON
OpenWindowTaglist() - when FOCUS feature or OpenWorkbench fix is ON
OpenWorkBench() - when OpenWorkbench fix is ON
LockPubScreen() - when OpenWorkbench fix is ON
CloseScreen() - when OpenWorkbench fix is ON

ExecBase

ex_LaunchPoint

1.14 Installed files

INSTALLED FILES

~~~~~

All Executive files, except a locale catalog are installed to a single directory. The preferences file, "Executive.prefs" is initially written to this directory, but Executive will also search it from these directories:

```
current directory
PROGDIR: (directory where the program binary is)
ENV:
S:
```

These directories are also searched for the keyfile, "Executive.key".

A preferences file containing all the defaults is written during installation to the install directory. It also contains the BASEDIR option:

```
BASEDIR      "WORK:Executive"
```

This is the directory where Executive was installed. The AmigaGuide document file is searched from this directory when you start a client from CLI with the HELP option or press the HELP-key when using a client with a GUI. These directories are also searched for the documentation file:

```
PROGDIR:Help/<language>/
HELP:<language>/          (AmigaDos V39+)
PROGDIR:
Current directory
```

The locale catalog is installed if you are using AmigaDos 2.1 or newer. AmigaDos 2.04 doesn't support localization. The catalog is installed to LOCALE:Catalogs/<language>/Executive.catalog.

To uninstall Executive click on the UnInstall icon and follow given instructions. This will safely remove Executive from your harddisk.

### MOVING EXECUTIVE TO ANOTHER DIRECTORY

~~~~~

Experienced users may want to move Executive to another directory. Here's how to do it:

```
Copy the server and clients to a directory that is in your path,
for example C:.
```

```
Copy the keyfile and the preferences file to your S: directory.
```

Copy the AmigaGuide manual to HELP:<your default language>/ directory.

Remove the BASEDIR line from the preferences file.

Remove the Path-command from S:User-Startup file and change the path to the Executive binary.

If you don't use Workbench, you can delete all the icons.

Of course the supplied UnInstall script can't be used anymore, so you can delete it.

1.15 Common problems

COMMON PROBLEMS

Here are a few common problems and solutions for them. These are not bugs, but "features" of Executive.

ERRORS IN SERIAL TRANSFER

If a communication program is scheduled by Executive, its priority may drop too low and then it gets blocked by higher priority tasks and won't receive enough CPU time, so characters gets lost. Communication programs usually use a lot of CPU time, so Executive will lower their priority, especially during up/downloads.

This problem is easily solved. Start

ExecutivePrefs

and go to the tasks-

section. Press the New-gadget to create a new entry for a task that the server should watch. Press the Magic Wand-gadget and see if the communication program you're using is on the list. If it is, just select it and press Ok. Press the Ok-gadget in the Edit-window too and save the preferences. Now you must restart Executive so the changes take effect.

Note that you have to start the communication program from Workbench, or use RUN-command from CLI, if the communication program doesn't automatically detach itself from CLI. If you just execute the program in your current CLI, no new task is created and Executive won't notice the program at all.

If the communication program that you're using isn't on the Magic Wand-list, then choose "Communication program" from the list and press Ok. Start the communication program so you'll see what's the name of it's task. Press the Name-gadget (in the MUI version of ExecutivePrefs this is a small popup-gadget) and select the communication program from the list, press Ok and save the preferences.

It's possible that the communication program creates several tasks. In this case the all the tasks names usually start with the same string, for example "Term", so type "Term*" in the name-gadget.

Leave all other settings untouched (i.e. TASK, NOSCHEDULE and ABOVE selections should be activated).

TERM (by Olaf Barthel)

If you have added Term to the watched tasks list so that it should not be scheduled, Executive won't notice it because Term has a small bug that causes the CLI command name to be incorrect. I have sent a fix for this bug to the author, so it will hopefully be fixed soon.

To set Term so that it's not scheduled, all you need to do is to go to the Miscellaneous settings-section in term and change Term's priority above the catch range. If you are using the default settings then just change Term's priority to 3.

MAGIC WANDS

There are a few programs that shouldn't be scheduled. ExecutivePrefs has preset settings for these programs, you only need to copy them from the Magic Wand list to the Executive.prefs-file. See

ExecutivePrefs
for how to use the Magic Wand list.

When you installed Executive your system was scanned for programs that are in the Magic Wand list and they were automatically added to the preferences file. This only works for programs that were running at the time you installed Executive, others must be added manually.

All communication programs must not be scheduled, for reasons given above and their priority must be kept above the catch range.

Screenblankers that use a lot of CPU time when the screen is blanked, should not be scheduled so they won't steal CPU time from other programs. Their priority should be kept below the catch range.

Some multi-function commodity programs like MagicCX and MCP have features (like SunMouse on MCP) that don't work if they don't get CPU time almost immediately they need it. Although the scheduler keeps these tasks running at a very high priority (because they don't use much CPU time) it's still possible that they become blocked by some higher priority task. So we just set them so that they are not scheduled and their priority is left untouched. These programs should have a higher priority than 0 or 1 even when they are used without Executive.

ShapeShifter, the Macintosh emulator requires that it's maintask's priority is kept below the support tasks' priorities. It's ok to schedule the main task, but the support tasks must not be scheduled.

If you like fast response from ToolsDaemon, it's best not to schedule it. You CAN schedule it but then it won't respond as fast as it would if it's not scheduled.

Brilliance is programmed in such a way that changing its subtasks' priorities causes some mouse and keyboard input to be ignored.

CHOOSING THE RIGHT TIMER

After you have installed Executive, you might get a message from some programs that CIA-A or CIA-B timer is not available. Executive can use three different

```
timers
:
```

```
CIA-A
CIA-B
timer.device
```

Executive defaults to CIA-A on 68000/010 Amigas and to CIA-B on 68020+ Amigas. CIA-B timer may cause serial transfer errors on slower Amigas (nothing to do with Executive) and timer.device uses CIA-B too, so CIA-A is the best choice for 68000/010 Amigas.

CIA-B should be used on all other Amigas because it's bit faster than using timer.device.

ShapeShifter requires that you use CIA-B timer, because it can only use CIA-A.

MOUSE MOVEMENT IS SLUGGISH, SYSTEM SEEMS TO BE VERY SLOW

This happens if you try to schedule Workbench. The default is NOT to schedule Workbench, so if you must have set the SCHEDWB-option on. Use

```
ExecutivePrefs
to correct the problem.
```

EXECPATCH

If you're having problems like Top displaying 0% CPU usage, you might be using a hack called ExecPatch. I don't know what other problems it may cause because, surprise surprise, it hangs on my system.

OCTAMEDPLAYER & DELITRACKER & MULTIPLAYER

If you're using CIA-A timer with Executive, some versions of OctaMEDPlayer crash because they can't allocate a timer. Get the latest version of OctaMEDPlayer from Aminet, the file is mus/play/OctaMED-P.lha.

MultiPlayer's and Delitracker's MED-players crash if you use CIA-A timer with Executive. Use ExecutivePrefs to change the timer to CIA-B or timer.device, both will work fine.

QUADRACOMPOSER

This program doesn't know how to use CIA-B
 timer
 if CIA-A is allocated.
 Set Executive to use CIA-B timer or timer.device instead of CIA-A.

AIR

AIR is an infrared remote controller program from Geodesic Design. It uses the CIA-B

timer
 without allocating it from ciab.resource. If
 Executive is set to use CIA-B, AIR will mess up with Executive timing and crash is likely to occur. You have to tell Executive to use CIA-A timer or timer.device if you want to use AIR.

1.16 Background

BACKGROUND

~~~~~

Development of Executive was started when I had a debate about process scheduling on AmigaDOS and on OS/2. At the same time I also bought a bigger harddrive, so I could install NetBSD, the free UNIX operating system for Amiga. NetBSD has a "real" scheduler, and I wanted to have something similar in AmigaDOS.

So I started with the project. There weren't any similar programs available for Amiga, so I had to improvise. The first version used to calculate CPU usage by patching launch and switch vectors in every task structure. This was a bad idea and it wouldn't have worked with tasks that wanted to use these or the userdata field. So I changed the code so that CPU usage was calculated using a vertical blanking interrupt. I incremented a counter for each task if they were being run when the interrupt occurred. This worked and it was very efficient, but the accuracy was only 1/50 second, and in practice it was even less, because tasks that used CPU only between two vblanks didn't seem to use CPU at all. The current method for calculating CPU usage is explained elsewhere  
 , so I won't repeat it  
 here.

## 1.17 Server

### SERVER

~~~~~

```

Starting the server

Making it work

Options

Server internals

Quitting the server
    Start server

Stop server

```

1.18 Starting the server

STARTING THE SERVER

~~~~~

If you have installed Executive normally, it will be started when your system is booted.

It's possible to start Executive anytime you want to, but then it's not possible to resolve existing child-parent relationships so the

```

Pstree
-client can't display them.

```

Also, if you want the focus-feature to work properly, you should start Executive before any windows have been opened.

The server can be started from Workbench or CLI. It must be 'run' from CLI, so use this command to start it:

```
run <NIL: >NIL: Executive
```

## 1.19 Making it work

### MAKING IT WORK

#### PRIORITY RANGES

~~~~~

Every task in Amiga has a priority which varies from -128 to +127. The priority is fixed and is usually only changed by the user. Executive will change priorities automatically and give lower priority to tasks that want to use much CPU time and higher priority for tasks that use less CPU time. Executive reserves a range of priorities called dynamic range where it places the scheduled tasks. The default range, -50 to -100 is normally unused, so it's a good place, but a different range can be

specified.

The dynamic range MUST be inside another priority range, called catch range. By default, the catch range is from +2 to -105. Every task that is started, or whose priority is later changed to a value in this range, will be scheduled. If some task is being scheduled and its priority is changed to some value outside the catch range, it won't be scheduled any more.

This is a crude map of the priority ranges explained above:

```

+127 - +
+126 - | Executive server runs at this priority, +126.
      |
      |
      |
      |
      |
      |
+2  - | <-----'.
0   - +                               | By default, tasks with
      |                               | priority from +2 to -105
      |                               | will be 'caught' by the
-50 - | <--. Scheduled tasks will be | server and they will be
      |   | assigned a priority from | scheduled.
      |   | this area, the dynamic range, |
-100 - | <--' by default from -50 to -100. |
-105 - | <-----'
      |
-128 - +

```

The maximum dynamic and catch range is from -125 to +125.
The dynamic max - dynamic min must be at least 2.

It's possible to schedule ALL tasks in your Amiga, i.e. set catch range to -125 - +125, but usually tasks with priority 4 or higher work more efficiently when they are not scheduled.

If you don't use any of the schedulers, then you don't have to care about any priority ranges.

Why it's necessary to run the server with so high priority? Is it dangerous? No, it isn't. The server needs to be run with a priority above all other tasks so it can always be run when it needs to, otherwise CPU usage calculation would not be correct. Please don't run any other tasks with priority 126 or higher, it will affect CPU usage calculations.

WATCHED TASKS

Usually you want to schedule all tasks that have their priority in the catch range. But there are some programs that shouldn't be scheduled for certain reasons. For example, a telecommunication program (NComm, Terminus, Term, ...) shouldn't be scheduled because its priority may drop below some other time consuming tasks and then it wouldn't get all the CPU time that it needs. This may cause transfer errors. So you must run this task with a priority above the catch range, or

tell Executive not to schedule this task, even if its priority is in catch range. Task's priority won't be changed, if it isn't in the dynamic range, where the scheduled tasks are. If task's priority is in the dynamic range, its priority will automatically be raised above the dynamic range.

If the telecommunication program has an option to set its priority, you should use it. If no such option is available, then tell Executive to handle it (this will be explained later). If the telecommunication program is not started from Workbench or RUN from CLI, Executive won't notice it, because a new task won't be created.

So, Executive watches new tasks and checks if their name matches a task in its watched tasks list. If it does, some of these actions might be taken:

- * Keep the priority of a task above or below the dynamic range and don't schedule the task.
- * Schedule a task even if its priority is not in the catch range. You can also specify a default nice-value for this task.
- * Don't schedule a task but set its priority to a specified value. The priority can be changed afterwards. You probably won't need this.

These actions can be applied to the specified task, its childtasks or both.

Sometimes when a process (not a task but process) is created (e.g. with RUN command from CLI), it's possible that the CLI command name is not available immediately upon invocation. Executive will wait for approximately 1.5 seconds before it checks if this process is being 'watched', the name should then be correct. After approximately 10 seconds has passed since the process was create, Executive will check the name again if necessary.

Task names are case insensitive, and you can use * as a wildcard. For example, '*ed' will match all task names that end with 'ed'. If the task is a CLI process, then the CLI command name will be used.

Workbench is a special task that shouldn't be scheduled, because it would make Workbench rather unresponsive. This is built-in, so you don't have to worry about it. It's still possible to schedule the Workbench task, there's an option to do this.

Rarely it's necessary to totally ignore a task. Currently I know of only one task that must be ignored. The SAS/C Development System has a debugger called CPR which creates a task called 'VISOR input processor'. This task is not removed in a legal way (RemTask() is not called), so it will cause problems with programs like Executive. But this will be handled AUTOMATICALLY by Executive, so there's no need to worry about it.

There's absolute no need to ignore any other tasks at the moment, but it's possible. If you ignore a task, then Executive clients don't know anything about it.

1.20 Server options

SERVER OPTIONS

~~~~~

Before parsing command line options or tooltypes, the server reads a preferences file. The easiest way to change the preferences file is to use the

ExecutivePrefs  
utility. If you want to edit preferences  
file by hand then please read the  
preferences file  
documentation.

It's not possible to change any options on the fly, you must always restart the server.

If the server is started from Workbench, tooltypes will be read, otherwise CLI arguments are parsed. The server understands these CLI arguments and tooltypes:

#### CATCHMAX

Template: CATCHMAX/N/K  
Tooltype: CATCHMAX  
Default: 2

The highest priority a task can have to be caught by the server and scheduled.

#### CATCHMIN

Template: CATCHMIN/N/K  
Tooltype: CATCHMIN  
Default: -105

The lowest priority a task can have to be caught by the server and scheduled.

#### DYNAMICMAX

Template: DYNAMICMAX/N/K  
Tooltype: DYNAMICMAX  
Default: -50

The upper bound for dynamic range.

#### DYNAMICMIN

Template: DYNAMICMIN/N/K  
Tooltype: DYNAMICMIN  
Default: -100

The lower bound for dynamic range.

#### NOFOCUS

---

Template: NOFOCUS/S  
Tooltype: NOFOCUS  
Default: Focus option is ON by default

If you specify this switch, the focus-feature is disabled. By default, the task that owns the active window gets more CPU time.

#### SCHEDWB

Template: SCHEDWB/S  
Tooltype: SCHEDWB  
Default: Workbench is not scheduled by default

Workbench is not scheduled, because that would cause noticeable slowdown in performance. Use this switch to schedule Workbench.

#### DONTFIXOPENWB

Template: DONTFIXOPENWB/S  
Tooltype: DONTFIXOPENWB  
Default: Fix OpenWorkbench bug

There's a bug in AmigaOS 3.x which causes system to lock up if a task that opens the Workbench screen has a priority lower than 1. Executive fixes this bug by raising task's priority to catchmax+1 or to at least 1. Task's priority will be set back to normal after the buggy system function has been executed. You can disable this with this option.

There are some utilities like MultiCX and MCP that also know how to fix this bug. They set task's priority to 1, but if you are running Executive, this has no effect. If this priority is inside the catch range, Executive will schedule the task and its priority won't be changed. You should disable the fix OpenWorkbench bug feature from these utilities if you're using Executive.

#### TIMER

Template: TIMER/K  
Tooltype: TIMER  
Default: CIA-A (on 68020+ Amigas CIA-B)

Which

timer  
to use:

CIA-A  
CIA-B  
DEVICE (timer.device)

#### SCHEDULER

Template: SCHEDULER/K  
Tooltype: SCHEDULER  
Default: STANDARD

---

Which scheduler  
to use:

SIMPLE  
STANDARD  
MARKET  
REMEMBER  
QUEUES  
SUPER

Specify NONE if you don't want to use any scheduler.

## QUIT

Template: Q=QUIT/S  
Tooltype: QUIT

This will tell the server to quit and remove itself. For more  
information about quitting the server, see  
here  
.

## HELP

Template: HELP/S  
Tooltype: HELP

Display on-line help-file for the server.

## VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.21 Quitting the server

### QUITTING THE SERVER

~~~~~

You can quit the server by issuing this command:

Executive QUIT

or run the 'Executive' -program from Workbench and you'll be asked if
you want quit Executive.

When the server quits, it will first signal
Acct
to write all unpurged
accounting records to disk.

You might get this message when you try to quit the server:

```
Unable to quit. Some program has patched a
function in Exec or Intuition library.
```

This means that you have run a program after the Executive server which has patched one or more of the same functions as the server.

The patches must be removed in reverse order, so you must check what program might have installed the patches. If this happens always when you try to quit Executive, then you should change the order of the programs you run in s:startup-sequence and s:user-startup. This requires some knowledge, so don't touch these files if you aren't sure what you are doing.

Another way to solve the problem is to use a program called SetManager by Nicola Salmoria.

You might also get this message while trying to quit the server:

```
Unable to quit. Some clients are still running.
```

You must quit all clients before quitting the server. You might have left

```
Top
or
ALoad
running.
```

1.22 Preferences file

PREFERENCES FILE

The preferences file is called 'Executive.prefs'. It's searched from the following directories in this order:

```
current directory
PROGDIR: (directory where the server binary is)
ENV:
S:
```

This is an ASCII file and contains one option on each line. Blank lines and lines starting with #-character are ignored. The option name and the value must be separated by spaces or tabs. Option names are case insensitive.

These options work like the CLI arguments and Workbench tooltypes, so consult their

```
documentation
.
```


CATCHMAX
 CATCHMIN
 DYNAMICMAX
 DYNAMICMIN
 NOFOCUS
 SCHEDWB
 TIMER
 SCHEDULER

These options are used only in the preferences file:

TASK
 CHILDTASKS
 TASK+CHILDTASKS
 IGNORE
 BASEDIR

TASK

```
Template:  TASK <task name> <NOSCHEDULE>  [priority]
                                                [BELOW]
                                                [ABOVE]
                                                <SCHEDULE>  [nice-value]
```

This tells the server to watch for tasks with the given name. The name is case insensitive and * can be used as a wildcard.

When a task matching the name is found, it's either NOT SCHEDULED or SCHEDULED. If you want the task NOT to be scheduled, you can optionally tell the server to keep the task's priority at the given value, or BELOW or ABOVE the dynamic range.

If the task is to be scheduled, you can give a default nice-value for it. Otherwise task will inherit its parent's nice-value.

CHILDTASKS

```
Template:  CHILDTASKS <task name> <NOSCHEDULE>  [priority]
                                                [BELOW]
                                                [ABOVE]
                                                <SCHEDULE>  [nice-value]
```

Similar to TASKS option, but applies to all childtasks created by the named task.

TASK+CHILDTASKS

```
Template:  TASK+CHILDTASKS <task name> <NOSCHEDULE>  [priority]
                                                [BELOW]
                                                [ABOVE]
                                                <SCHEDULE>  [nice-value]
```

Similar to TASKS and CHILDTASKS options, but applies to both the named task and all its childtasks.

IGNORE

Template: IGNORE <task name>

Tell the server to totally ignore the given task. Wildcards are supported and the name is case insensitive.

IMPORTANT! The <task name> is the actual task name, not the name of a CLI command. If task must be ignored, the server must do it at the task creation time and the CLI command name is not yet available.

This is needed only when a task is not removed with RemTask(). This is EXTREMELY rare and, illegal.

BASEDIR

Template: BASEDIR <directory>

This option is generated by the installation program. It's used when displaying on-line help so don't change it or delete it.

When you save the preferences file from
 ExecutivePrefs
 you'll lose the comments
 you have in the preferences file.

Here's an example preferences file:

```
#### install directory (please don't delete this) ####
BASEDIR    "SYS:Executive"

#### catch- and dynamic ranges ####
CATCHMAX   2
CATCHMIN   -105
DYNAMICMAX -50
DYNAMICMIN -100

#### options ####
#NOFOCUS
#SCHEDWB
SCHEDULER  SUPER
TIMER      CIA-B

#### tasks ####
TASK       Top          SCHEDULE   -15
CHILDTASKS ASwarm      NOSCHEDULE  BELOW
TASK       CygnusEd     SCHEDULE   -20
```

1.23 Server Internals

SERVER INTERNALS

~~~~~

---

## SCHEDULING

Executive schedules tasks by recalculating their priorities. This is done once a second. Priorities are calculated only if the task has recently used CPU time.

## PID

PID means Process IDentifier. Every task has a unique PID, allocated by the server. If task is a CLI, the PID will be the CLI number. Otherwise a unique number from 200 upwards is given to the task.

PIDs are very useful when you need to change a nice-value or priority of some task. You don't have to type in the name and sometimes there might be many tasks with the same name.

## NICE-VALUE

Each task has its own nice-value, usually 0. Negative nice-values (up to -20) give more CPU time to a task, i.e. it will be less nice to other tasks. Positive nice-values (upto +20) give less CPU time to a task. If you have two tasks that want to use all CPU time they get, and they have nice-values -20 and +20, the task with the +20 nice-value will still get some CPU time.

It's not possible to give any exact numbers of how much different nice-values have effect on task's priority. Each scheduler uses nice-values in somewhat different way and many other values affect the calculated priority.

## LOAD AVERAGES

Executive maintains three load averages, for the past 1, 5 and 15 minutes. A load average is the number of tasks that have been ready to run or running in the past 1, 5 or 15 minutes. If the load average is 1.00, you have had exactly one task running.

## WHAT TASKS SHOULD/SHOULDN'T BE SCHEDULED

There are very few tasks that shouldn't be scheduled, for example input.device and telecommunication programs. But usually you don't want to schedule any tasks that normally have a priority 3 or above.

If you have a screenblinker that draws some fancy graphics on the screen while the screen is blanked, it's better not to schedule it. You should lower its priority below the catch range, so it doesn't consume any valuable CPU time.

Some multi-function commodity programs like MagicCX and MCP have features (like SunMouse on MCP) that don't work if they don't get CPU time almost immediately they need it. Although the scheduler keeps these tasks running at a very high priority (because they don't use much CPU time) it's still

---

possible that they become blocked by some higher priority task. So we just set them so that they are not scheduled and their priority is left untouched. For most common programs this is done automatically during installation. These programs should have a higher priority than 0 or 1 even when they are used without Executive.

#### FOCUS

Usually the task that has opened the currently active window is the one that receives user attention. Executive will give more CPU time to this task if you haven't turned the focus-feature off. The exact amount varies, depending on the scheduler you have chosen.

When you start the server, it will scan opened windows trying to figure out their owner, but it's not always possible, so it's best to start Executive before any windows have been opened. You can see the number of windows whose owner is known by using the

```
Stat
client.
```

#### CHILD-PARENT

Executive maintains information about each task's parent and its children. It's not possible to know this relationship for tasks that were in the system when the server was started.

If a parent of some task terminates, its children will be transformed to the parent of the terminated task.

#### PERFORMANCE

The server uses very little CPU time. Top and Ps show how many percentages of CPU time the server actually uses. Some of this time goes to serving clients. If you start 10 ALoad clients, you can see a tiny increase in server's CPU usage.

As a technical detail, Executive uses hash-tables to look up the internal information structure for each task, so there is no need to traverse long task lists.

So all in all, the cost of using Executive is insignificant. You'll easily spend more time changing priorities than Executive uses in a few weeks.

#### REAL PRIORITIES

Executive keeps track of the real priorities of all tasks so it can restore them when you quit the server. The real priority is the one that is set by SetTaskPri() call.

#### STRANGE PRIORITIES

---

Exec.library lacks a function GetTaskPri(), which could be patched so the real priority of the task could be returned. Now it's possible that some task reads its 'scheduling' priority and starts a childtask with this priority. The server will notice this and replace the priority with correct value. This must be done, because if you will quit the server, the childtask's priority would be incorrect.

Sometimes a childtask is started with parent priority - 1 (for example). The server can't correct this, but this is not a problem unless you quit the server. Then the task's priority will be incorrect, usually some negative value in dynamic range. But this is really very rare.

#### CONTEXT SWITCHES

Executive keeps track of the number of context switches for each task. There are two types of context switches, voluntary and involuntary. Voluntary context switch happens when a task calls the Wait()-function and is put to sleep. Involuntary context switch happens when a task is forced to give the CPU to another task.

#### TIMERS

Executive needs to be signalled by a timer every second so it can recalculate priorities. Amiga has a device called 'timer.device' which can be used to get accurate timing signals. There's just one problem: timer.device may cause characters to be lost during serial transfer because the serial interface in Amiga is not buffered.

There are two kinds of hardware timers in Amiga, CIA-A and CIA-B. Timer.device uses CIA-B. When a certain time has elapsed, the CIA timer causes an interrupt. CIA-B uses higher level interrupt than the serial interface, so it's served first. If the serial interface interrupt can't be served in time, a character is lost. CIA-A uses a lower level interrupt than CIA-B and the serial interface.

On Amigas with 68000 or 68010 Executive first tries to allocate CIA-A timer. On Amigas with 68020+ Executive defaults to CIA-B. If the default timer isn't available, Executive will try the other one. If this fails, Executive uses timer.device.

You can change the preferred timer using the TIMER-option. If you don't have any problems with serial transfers, you may want to use CIA-B because then CIA-A timer will be left for other programs. For example, HippoPlayer and ShapeShifter want to use CIA-A timer, so use CIA-B or timer.device for Executive.

To see what timer Executive is using, use the  
Stat  
-client.

#### OPEN WORKBENCH BUG FIX

See

DONFIXOPENWB option

---

for more information about this.

IS IT A HACK?

No, Executive is not a hack. The only 'illegal' thing that must be done is to patch the undocumented Switch()-function in exec.library and to install a routine to ExecBase->ex\_LaunchPoint. These are needed to get accurate CPU usage for tasks and it's the best way to do it. There's no way to get more accurate CPU usage values on AmigaDos.

HOW THE CPU USAGE FOR A TASK IS MEASURED?

The CPU usage calculation starts when a task is launched by the dispatcher and it's stopped just before the task is switched to another. The task time doesn't include the time that is spent in exec.library for saving the task's registers and finding a new task to run during a context switch. That's why the

Top client rarely shows 100% CPU usage. If it's 99.7%, then the 0.3% is spent in the task dispatcher doing 'unproductive' work.

## 1.24 Clients

CLIENTS

~~~~~

Clients are utility-programs that are used together with the Executive server. Most of the clients require the server to be running to work.

Here is a quick overview of included clients:

Acct	Accounting daemon
ALoad	Display load average or CPU usage
ALoad3D	Display load averages in 3D
Commander	General task manager
Ctp	Extended ChangeTaskPri
Kill	Extended Break
Lastcomm	List last commands executed

Meter	Display system information
Nice	Run programs with lower scheduling priority
Ps	Process status displayer
Pstree	Display child-parent relationships
Renice	Renice a task
Sa	Display accounting statistics
Stat	Display some misc. information
Timer	Time CLI and Workbench programs
Top	Display information about the top CPU tasks
Uptime	Display system uptime and load averages

All clients can be run from CLI and Workbench. Most of the options can be given as CLI arguments or Workbench tooltypes.

You can always use the HELP option or a tooltype with the same name with all clients and get to their help-page.

Each client has an option called VERSION, which can only be given from CLI. It displays the client's version number.

When you run a program from Workbench, the program opens a window for its output if necessary. The default size for this window is 640x200 pixels, but it can be changed with an environmental variable EXECUTIVEWINDOW. For example, if you have a PAL machine, you want a window which is 256 pixels high. You can set the environmental variable to the desired size with these CLI commands:

```
setenv EXECUTIVEWINDOW "0/0/640/256"  
copy ENV:EXECUTIVEWINDOW ENVARC:
```

The format of the environmental variable is left/top/width/height.

CLI ARGUMENTS

When a client is invoked with a question mark as an argument, it will output the argument template. For example, the command "Ps ?"

will output this:

```
F=FORMAT,S=SORT/K,LIST/S,NH=NOHEADER/S,PH=PADHEX/S,PLAIN/S,DELAY/N/K,
IV=INVERSE/S,ACTIVE/S,IDLE/S,SCHED=SCHEDULED/S,RUN/S,READY/S,WAIT/S,
ALL/S,OWN/S,P=PID/K,U=USER/K,G=GROUP/K,N=NAME/K,R=PGRP/K,PRIGT/N/K,
PRILT/N/K,PRIEQ/N/K,NPROCS/N/K,EVERYTHING/S,BABY/S,TASK/S,PROCESS/S,
CLI/S,CPU/S,HELP/S,VERSION/S:
```

A CLI argument is divided in three parts:

```
NH=NOHEADER/S
^^ ^^^^^^^ ^
| | |
| | | `-- modifier
| `-- option name
`-- can be used instead of the full name
```

The modifiers used in Executive are:

/S - Switch. For example, "Ps SCHEDULED" will output only the scheduled tasks.

/N - A decimal number.

/K - Keyword. This means that the option will not be understood unless the option name appears. For example if the template is "PGRP/K", then unless "PGRP=<pgpr>" or "PGRP <pgpr>" appears in the command line, PGRP will not be accepted.

/A - Required. This keyword must be given a value during command-line processing, or an error is returned.

/F - Rest of line. If this is specified, the entire rest of the line is taken as the parameter for the option, even if other option keywords appear in it.

If there are no modifiers after the option name, it's an string argument which can be specified without the option name. So you can invoke Ps with a format string in three ways:

```
Ps FORMAT=pid,cpu%,name
Ps F=pid,cpu%,name
Ps pid,cpu%,name
```

1.25 Schedulers

SCHEDULER

Scheduler is a mathematical algorithm that calculates priorities for scheduled tasks. Schedulers use different kinds of information when calculating priorities:

* load average

- * task's recent CPU usage
- * task's context switches

Let me first say that there's no need to use a scheduler on Executive, but you'll probably want to use one of the 6 different schedulers available. All schedulers on Executive are fair-share schedulers, they'll make sure that each task gets a chance to run, but favour tasks that only use a little CPU time, i.e. keep compute-bound tasks at low priority and transput-bound tasks at high priority. If the

focus-option
has been turned

on, the task that owns (i.e. has opened) the currently activate window gets a higher priority. Also, task's nice-value affects priority calculation.

If you aren't using a scheduler, then the priorities won't be changed by the server, but you can still use all other Executive features normally.

Schedulers work by calculating task priorities for scheduled tasks once a second. If a scheduler notices that a task hasn't used CPU time recently, it will stop calculating that task's priority.

When a new task is added, it will be given some of its parent's scheduling information (e.g. recent CPU usage) and when a task exits, some of its scheduling information is transferred back to the parent. This is especially useful when a task does most of its hard work in subtasks that it creates (AdPro for example uses subtasks for loading and saving images and when processing them).

To see how the system behaves under different schedulers use the
Ps
and

Top
clients and look how much CPU time different kinds of tasks get.

There are currently six different schedulers available:

- * simple
- * standard
- * remember
- * market
- * queues
- * super

See the

server options
documentation for information about changing

the scheduler. Please note that it's not possible to change the scheduler on the fly, you must restart the server.

I'll now outline some of the main differences between the schedulers:

SIMPLE

As the name says, this is the simplest of all the schedulers.

The calculated priority for a task is proportional to the task's recent (past 5 seconds) CPU usage. This scheduler is useful in very lightly loaded systems. Actually, load average 2.0 is used constantly.

STANDARD

This scheduler can handle much higher loads than the simple-scheduler. It's similar to the simple-scheduler except that it also considers the load average when calculating priorities. When the past minute load average gets higher, used CPU time will 'cost' much more.

REMEMBER

This is a 'heavy-duty' scheduler, it can handle very highly loaded systems. It's similar to the standard scheduler, but it will remember a lot longer when a task uses much CPU time and it will forget small CPU usage very quickly.

MARKET

This scheduler is based on 'bidding' for CPU time. Each task gets some new money from the scheduler every second. Used CPU time will cost money, and the task that has most money gets the first chance to run, i.e. the highest priority. When the system load average is higher, CPU time will cost much more.

QUEUES

This scheduler divides the dynamic range to eight priority levels and runs several tasks with the same priority. Tasks that use approximately the same amount of CPU time get same priority.

Exec will Round-Robin tasks that have same priority. Each task is run for a time-slice called Quantum and then the next task gets chance to run. Exec normally uses a fixed Quantum value 4, but this scheduler adjusts the Quantum value for each task to maximize throughput of CPU-intensive tasks and to make interactive tasks more responsive. Here are the Quantum values given to tasks with different priorities:

PRIORITY	QUANTUM
above dynamic range	4
scheduled (inside dynamic range)	2-10
below dynamic range	12

For scheduled tasks the Quantum is largest (10) when task's priority is close to dynamicmax priority (by default -100).

This scheduler tries to eliminate "wobbling" priorities by not letting a task migrate upwards immediately. Task has to wait for one second before its priority can be raised.

This is currently the best scheduler in Executive.

SUPER

This is similar to the standard-scheduler, but it also uses

```
context switch counters
when calculating priority for a task.
```

Tasks that use a lot of CPU time must usually be forced off the CPU involuntarily, these tasks will be given lower priority than tasks that voluntarily give the CPU to another task. The result is that transput-bound tasks get more CPU time than compute-bound tasks.

In some cases even the SUPER scheduler can't handle the load because a lot of small tasks are created for a very short time. The best thing you can do is to renice the whole process group to +20. This kind of situation is pretty rare so don't worry. When I do some version management for Executive sources, I run a program called Make recursively for many subdirectories and

```
Pstree
outputs something like this:
```

```
+---[Make]
  +---CON
    +---[Top]
      `---[sh]
        `---[sh]
          `---[Make]
            `---[sh]
              `---[sh]
                `---[Make]
                  `---[sh]
                    `---[Make]
                      `---[Make]
```

The Make at the end is usually run for less than one second so sometimes a priority can't be calculated for it before it exits.

1.26 Accounting

```
ACCOUNTING
```

Most multiuser UNIX systems are usually set up to use accounting. The system administrator is able to see what programs have been run, who has run them and how much CPU time they have used. Accounting is very useful on single-user systems too. Have you ever wondered where all that CPU time goes to?

When accounting has been enabled, server will create an accounting record every time a program started from CLI or Workbench terminates. The accounting record consists of this information:

- Command name
- Time the command terminated
- Used CPU time
- Elapsed time
- User ID (if MultiUser has been installed)

- Group ID (if MultiUser has been installed)
- Miscellaneous flags

When enough accounting records have been created, or the oldest record has been in memory for a user specified time, the server signals

Acct

-

client and tells it to write the accounting records to disk.

Acct writes accounting records to a file called 'acct'. To see the last terminated commands, you use the

Lastcomm

-client. The most recently

terminated command is displayed first.

Because the 'acct'-file may grow rather large in a short time, it's necessary to condense this information in to a much shorter format and that is done by the

Sa

-client. Sa stands for summarize accounting.

Sa reads the acct-file and generates a summary which it writes to files called 'acct.procsun' and 'acct.usersun' which contain statistics according to command name and user ID, respectively. If these files already exist then they are read and included in the summary.

This information is included in the summary files:

- Command name (acct.procsun) or user ID (acct.usersun)
- Number of times the command has been executed (acct.procsun) or number of commands the user has executed (acct.usersun)
- Total used CPU time
- Total elapsed time

If MultiUser hasn't been installed, 'acct.usersun' will contain only one user.

Normally Sa will just display the summary and not write it back to disk. To write the summary to disk, you must use the MERGE-option with Sa.

Accounting records can't be created for any custom shell's (CSH, SkSH) internal commands.

1.27 Genies

GENIES

~~~~~

Genies are project icons that execute a client with some tooltypes. You can find them in the Genies-drawer. If you want to create your own Genie, just make a copy of one of the Genie-icons and change the default tool and tooltypes. For example, if you want to make a Genie for 'Pstree', set the default tool to '/Pstree'.

The genies won't work if you're using a utility that changes the

"/" in pathnames to ":", giving UNIX-like paths.

#### USING NICE FROM WORKBENCH

You can run programs with a different nice-value by creating a Genie for the program you want to run. Use

```
/Nice
```

```
as a default tool for the project
```

icon and put the name of the program you want to run to a tooltype COMMAND, for example:

```
COMMAND=c:VirusChecker DH0:
```

#### AVAILABLE GENIES

Here are the available Genies and their equivalent CLI commands:

##### Scheduled tasks

CLI:

```
Ps
```

```
SCHEDULED
```

Display scheduled tasks.

##### Merge accounting

CLI:

```
Sa
```

```
MERGE
```

Summarize

```
accounting
```

```
and merge it with the (existing) summary database.
```

The current accounting file will be deleted. The summary is displayed on screen.

##### User accounting

CLI:

```
Sa
```

```
USER
```

Display per-user accounting summary. If you're not using MultiUser, this will just display all commands grouped together.

## 1.28 Glossary

---

## GLOSSARY

~~~~~

blocked

A task that is ready but not running is blocked. This happens when a higher priority task is currently using CPU.

compute-bound

A task that spends most of its time just using the CPU without waiting for any external events is compute-bound. Opposite of transput-bound.

co-operative task-switching

See also: History.

catch range

Every task that has priority in this range is automatically scheduled by Executive. If task's priority is later changed to this range, it gets scheduled.

context switch

A context switch happens when the running task is switched to another. The context switch is involuntary if task didn't call Wait(), i.e. it had to be forced off the CPU.

dynamic range

When Executive calculates priorities for scheduled tasks, the priorities will be in this range.

Exec

Did you know that the full name of Exec is really The Multitasking Executive?

MultiUser

From the README-file included with MultiUser distribution:

MultiUser allows you to create a *IX-like environment where several users live together in harmony, unable to delete each others files, unable to read those private love-letters of other users. And this even if several users are working on the system at the same time (on a terminal hooked up to the serial port)

People without a valid login ID and password won't be able to access files you have made private with MultiUser. If you make all files private (not readable for others), the only useful thing they could do, is boot from a floppy.

You can get MultiUser from any Aminet site, see the util/misc-directory.

PID

Process Identifier. Each task is given a unique identifier by the server. When manipulation tasks from CLI, PID's are extremely useful.

preemptive

Preemptive multitasking means that the operating system can force a task to give the CPU to another task.

process

A process in Amiga is an extension of a task. The main difference is that only a process can use DOS to access files.

Round-Robin

The way the OS shares CPU time between tasks with same priority. Each task gets to run for a specified time called "quantum", then the next task with same priority is run and the current task is moved to the end of the list.

Simpson, O.J.

Actor.

task

On other operating systems than AmigaDos tasks are called processes, but on Amiga process has a special meaning.

transput-bound

A task is transput-bound if it spends most of its time waiting for some event to occur. Opposite of compute-bound.

1.29 Registration

REGISTRATION

~~~~~

UNREGISTERED VERSION OF EXECUTIVE

These features require a registered version of Executive:

- \* All schedulers except STANDARD
- \* Accounting
- \* ALoad3D
- \* Meter

The absence of these features doesn't make Executive unusable. You can use Executive without these features but I hope you'd register after you have seen what this program is.

WHAT DO I GET WHEN I REGISTER?

You'll receive a  
keyfile  
which enables the disabled features. You'll  
also ensure that all updates will be FREE for you. Currently I'm planning  
to distribute updates free of charge, but this may change in the future.  
Updates will always be free for registered users. Some new clients that  
I'm planning to add will be available for registered users only.

#### ORDERING

The easiest way to fill in a registration form is to use the "Register"  
program, which asks you all the necessary information and creates an  
ASCII registration form. A blank registration form is available in  
these formats:

FinalWriter  
PostScript  
DVI  
IFF  
ASCII

You'll find these files in the Register-directory.

If you don't have a printer, then create an ASCII registration form  
with the "Register" program and copy the necessary information to a blank  
paper. The information I need is:

- your name
- address
- email address
- payment method, currency
- method of delivery

You can also email me the registration form and send the payment by  
snail mail.

When you have filled in the registration form, please send it to me with  
your payment. My address is:

Petri Nordlund  
Vanhamaantie 4  
28800 PORI  
FINLAND

There's absolutely no need to pay extra for quick delivery, which seems  
to be common with some other shareware products. All orders are processed  
within 24 hours.

#### REGISTRATION FEE

Registration fee is accepted in these currencies:

15 USD (US dollar)  
10 GBP (Pound sterling)

---



70 FIM (Finnish mark)  
20 DEM (Deutsche mark)  
100 NOK (Norwegian krone)  
120 SEK (Swedish krona)  
500 BEF (Belgian franc)  
70 FRF (French franc)  
25 NLG (Dutch guilder)  
100 DKK (Danish krone)

REGISTER TOGETHER WITH YOUR FRIENDS AND GET A DISCOUNT!

3-4 registrations 20% discount  
5-7 " 30% "  
8+ 40% "

All registrations and payment must be sent together. Note that I can't exchange coins at the local bank, so if you are sending cash, you have to round the price upwards. Please don't hesitate to ask me if you are uncertain.

There's no need to send a registration form for each user, but the following information is needed:

- name
- address
- phone number
- email address (if you have one)
- delivery:
  - by airmail on a 3.5" disk
  - keyfile emailed to you

Keyfiles will be sent separately to each user.

#### PAYMENT METHODS

Cash This is the preferred payment method and also the cheapest for you and me. Make absolutely sure that the bills/notes are disguised. Enclose the cash in two pieces of non-white paper to disguise it. Please DON'T SEND COINS, they cannot be exchanged at the local bank.

Eurocheque Write the cheque for 70 FIM, that is, 70 Finnish marks. You must write your card number on the back of the cheque, otherwise I can't cash it.

Money transfer Transfer 70 FIM to my account:

- bank: Postipankki Ltd, Helsinki, Finland
- account number 8000 29-26522193
- pay via Eurogiro or SWIFT
- SWIFT address: PSPBFIHH; Telex 121 698 pgiro fi
- include a copy of the receipt with your registration form

This payment method is available for Finnish users only:

---

Postiennakko Postiennakkomaksu 25 mk lisätään hintaan (70+25=95 mk).

#### METHOD OF DELIVERY

If you have an Internet email address, I'll email you the keyfile on the same day I receive your registration. Otherwise I'll send you a 3.5" disk by post with the keyfile and the latest version of Executive.

The latest version of Executive can always be downloaded from Aminet. To find out what is the latest version, see the Executive WWW-page:

<http://www.megabaud.fi/~petrin/Executive.html>

Fidonet is too insecure for transmitting the keyfile.

## 1.30 Keyfile

#### KEYFILE

~~~~~

The keyfile contains your name, your address and a serial number. It's strictly forbidden to give this keyfile to anyone else. Several digital fingerprints are included in this file and redistributed copies are easy to identify.

When the Executive server is started, it will search for the keyfile from these directories:

```
current directory
PROGDIR: (directory where the server binary is)
ENV:
S:
```

If you want to put the keyfile to encrypted partition or just somewhere else than in these directories, put the directory name to environmental variable "KEYPATH".

1.31 Author

AUTHOR

~~~~~

Executive has been a large project and I have sometimes lacked motivation to continue with its development, considering the uncertain future of Amiga, but things look much better now.

It's always nice to hear comments and suggestions about this program, as those have already given me a lot of motivation and driven the project further.

---

I intend to continue development and release new versions regularly, so your thoughts and ideas are always welcome. Please tell me what you'd like to see implemented in Executive and what could be done better.

My Amiga is an old Amiga 2000, equipped with GVP G-Force 030/25 turbo, 340MB and 40MB harddrives and 10MB of memory. I also have a Citizen 120D printer and SupraFAX V.32bis modem.

You can contact me in several ways:

Email

petrin@megabaud.fi  
petrin@sik.ppoy.fi

WWW

Executive WWW page:

<http://www.megabaud.fi/~petrin/Executive.html>

My homepage:

<http://www.megabaud.fi/~petrin>

Snail mail

Petri Nordlund  
Vanhamaantie 4  
28800 PORI  
FINLAND

Telephone

+358-39-6480 322 (EET)

PGP

For secure communication, please use my  
PGP key

.

## 1.32 PGP KEY

PGP key

~~~~~  
Below you will find my public key. Use a text editor to save it to disk and follow the instructions on PGP manual for using it to send encrypted email.

You can also obtain my PGP key by fingering one of my email addresses.

-----BEGIN PGP PUBLIC KEY BLOCK-----

Version: 2.6ui

```
mQCNAjBz1LwAAAAEALP5+1K/Mqbr7VDFrsL/W3SrplhRfp5GbLid6Gn38/Zfb1PJ
hb/yDAuJX0G9nZ/hMBS7nXPVuUPZsfliEeOuDW6U3MiqOrhau7mi6P05KqTW/xYY
fzTeZ+K6relfoR0+ScV4i0fYpMi/O4n+3uZGVGc4X/JSVQyHsTDgjUTe/EdpAAUR
tCNQZXRYaSBOb3JkbHVuZCA8cGV0cmluQG1lZ2FiYXVklmZpPg==
=EBE0
-----END PGP PUBLIC KEY BLOCK-----
```

1.33 History

HISTORY

V1.00

~~~~~  
First public release.

V1.10

- ~~~~~
- \* Top-client now supports VMM. It displays available virtual memory if VMM has been installed.
  - \* Added ToolsDaemon And VLT/VLTjr to the Magic Wand list.
  - \* The server now defaults to using CIA-B timer on 68020+ machines and CIA-A timer on 68000 and 68010.
  - \* Added a Common problems section to docs.
  - \* A .Product-Info file is now included.
  - \* ExecutivePrefs[\_MUI] will now check for duplicate entries in task-list.
  - \* If a new entry is created in ExecutivePrefs[\_MUI] by pressing the New-gadget but then Cancel-gadget is pressed in Edit-window, the entry will be deleted.
  - \* When the initial preferences file is written, system task list is scanned for programs that are in the ExecutivePrefs Magic Wand list and these tasks will automatically be added to the preferences file.
  - \* Ps now has a new built-in format string called STACK which displays stack usage statistics for each task.

V1.20

- ~~~~~
- \* Meter is a new client which displays all kinds of system information in a small window. It very customizable and you no longer need ten different programs to show all the information you need. See documentation for more information. Meter requires a registered version of Executive.

- \* Scheduler changes:
    - o ALL SCHEDULERS
      - Nice-value now has much greater influence on task's priority.
    - o QUEUES
      - Reduced the number of queues from 10 to 8.
      - A task is not allowed to migrate upwards immediately, instead scheduler forces the task to wait for one second first. This eliminates priority "wobbling" which happens when two or more tasks using approximately the same amount of CPU time are fighting for the CPU.
      - This scheduler now adjusts Exec's Quantum-value when a task is launched. Lower priority tasks get higher Quantum, letting them run for longer time before switching to another task with same priority.
      - See documentation for more detailed information.
    - o SUPER
      - Remember task type (CPU intensive or interactive) for longer time
      - Task's type now influences task's priority even if task hasn't used any CPU time recently.
  
  - \* Added a Flush memory menu item to top. This flushes unused libraries from memory.
  
  - \* If a process has empty CLI command name (Term has if it's started from Workbench) Executive will now use task name instead when search for a task from watched tasks list.
  
  - \* ALoad[\_MUI] can now display current CPU usage instead of system load average. You can also specify which colors should be used and you can optionally have a clock in the window's titlebar.
  
  - \* Top can now be run in a shell-window, without opening an Intuition window, so you can run top from a remote shell. Use the STDOUT- option for this. There's also another new option, ONCE, which will just output the information to stdout and then exit.
  
  - \* Commander and ExecutivePrefs windows are now much faster and take less memory.
  
  - \* Commander now has a zoom-gadget and a new option, ZIP, for zipping the window when Commander is started.
  
  - \* If a program was set to be [not] scheduled, and it was started using Run, it was possible that Executive didn't notice the task. This happens because the CLI command name is not set immediately when a task is created. And under heavy system load this may take several seconds. I added a workaround for this problem, but there's no bullet proof way to fix this completely. It should now work well enough in all imaginable cases.
  
  - \* Added a fix for a bug in AmigaOS 3.x which causes system to lock up if a task that opens the Workbench screen has a priority lower than 1. Executive will now set task's priority to catchmax+1 while executing Intuition functions that could cause the Workbench screen to be reopened.
  
  - \* Added some information to the Command problems section in docs.
-

- \* If the update interval was more than 1 second, ALoad[\_MUI] and ALoad3D displayed an average of the loadaverage, they now display the current value.
  - \* Stat will now show how many accounting records has been created. Stat will no longer output "Accounting is activated" because this should be obvious if accounting statistics are shown.
  - \* Install-script will now update existing preferences file if it finds tasks that are in the Magic Wand list.
  - \* If ExecutivePrefs[\_MUI] finds the preferences file from ENV: then preferences will be saved to ENV: AND ENVARC:.
  - \* Added a note about Delitracker MED-player to the Common problems section in docs.
  - \* You can now use CTRL-C to stop pstree.
  - \* Added MagicCX, MultiCX, ToolManager, TrapDoor, MagicMenu, MCP and JamMail to the Magic Wand list.
  - \* Uptime is now calculated from eclock instead of RAM:-disk creation time.
  - \* Programs started using WBStart-Handler are now accounted properly.
  - \* Added portuguese catalog.
  - \* MUI list headers are now shown with highlight pen, which is usually white.
  - \* Removed text from top documentation that said it's possible that top may cause serial transfer errors. This is not true.
  - \* You can now use TASK+CHILDTASKS-option in the preferences file, instead of two separate options TASK and CHILDTASKS.
  - \* Fixed a small bug that caused a wrong item to be selected when double-clicking an item in ExecutivePrefs listviews.
  - \* Optimized the code that reads the preferences file, this made the server and ExecutivePrefs[\_MUI] a bit smaller.
  - \* Fixed a bug in ctp and kill-clients. If NAME, USER, GROUP or PGRP-options matched multiple tasks, it was possible that not all tasks were affected.
  - \* Registration fee is now accepted in DKK (Danish krone).
  - \* Money transfer to my bank account is now possible. See documentation or the registration forms.
  - \* ALoad and top now use pens from screen DrawInfo.
  - \* Top now has options for specifying background and text colors.
  - \* Added a zoom-gadget to the top-client. Pressing it causes the window
-

to be zipped so that only the header is displayed. There's also a new option, ZIP, for zipping the window when top is started.

- \* ALoad and uptime windows are now simple refresh windows, which take less memory than smart refresh windows.
  - \* Top will not allow a user (except the superuser) to break other users' tasks if MultiUser has been installed.
  - \* The clients that have a STAYFRONT-option will now check if the window is frontmost before trying to bring the window to front.
  - \* Changed the ShapeShifter entry in Magic Wand list so that the ShapeShifter main task is scheduled, but not it's childtasks.
  - \* Corrected a lot of spelling errors in the docs. :) Also made some small improvements here and there.
  - \* Fixed a bug which caused some memory to be trashed when a client or the server displayed the AmigaGuide document.
  - \* Fixed a bug which caused task name to be displayed instead of CLI command name if process has a CLI structure but no CLI number.
  - \* Commander and ExecutivePrefs now retain listview position when updating the list. This only worked with OS3+ before.
  - \* Fixed the COMMAND-keyword in ps.
  - \* All icons now have tooltypes with default values.
  - \* Added some words to the Glossary in docs.
  - \* When top erased a task that became idle, it sometimes outputted a few miscellaneous characters.
  - \* Added Brilliance to the Magic Wand list.
  - \* When Commander's window is zipped, it's width is halved.
  - \* Added a new option ANGLE to ALoad3D. You can now specify the initial rotation angle. It's now also possible to rotate the bar-chart using left and right cursor keys.
  - \* Added a note about Term to Common problems section.
  - \* ALoad3D's display is now rendered immediately after the window is resized.
  - \* Commander will now update the task list when it's window is unzoomed. Commander\_MUI does the same thing when it's uniconified.
  - \* Top will no longer look for VMM every second if it isn't in use when top is started.
  - \* Added DirectoryOpus to the Magic Wand list. This is not necessary, but DO works faster and is more responsive when it's support tasks are not
-

scheduled.

- \* ExecutivePrefs[\_MUI] now checks if the Executive.guide file is in BASEDIR. If not, it sets BASEDIR to the current PROGDIR: there when prefs are saved.
- \* The server now changes its priority to 126 after it has read the keyfile and preferences so no disk access is done at this priority.

### V1.30

- ~~~~~
- \* Removed some 68020+ optimizations from the server that were actually SLOWER than equivalent 68000 code.
  - \* Meter can now display number context switches during the last second.
  - \* Added Polish Install and UnInstall translations.
  - \* Registration price in Deutsche marks is now only 20 DM. It seems that 5DM notes are quite rare.
  - \* Discount in registration price is now available if three or more people register at the same time. See documentation.
  - \* The SPEED-option in ALoad3D is now used to set the rotation speed in degrees. Default is 5\textdegree{}
  - \* The SPEED tooltype in ALoad3D now accepts value 0.
  - \* A new option BATCH in Top disables reading keyboard when information is displayed in a terminal. Another new option UIDS displays UIDs instead of user names.
  - \* A new option ALL in Kill sends all signals, C, D, E and F to a task.
  - \* GUI based clients now have a new menu item, "Project->Save config". This will save current settings to the client's icon, including window position.
  - \* Mutually exclusive menu items now work correctly.
  - \* Modified the installation script so that it now works even if you're using a program that patches the dos.library so that you get UNIX-like path handling. For example, Yak with the UNIX Dirs-option turned on did this and caused the installation not to work correctly.
  - \* A new option MAX in Top displays as many tasks as possible.
  - \* When Top is running in a terminal, "n" can be used to change the number of tasks that are displayed.
  - \* ALoad3D now displays load averages more like ALoad. For load average 1.0 the bar is as high as possible. When the load average grows, the bar is divided in two parts. When the load average is for example 3.5, the bar is divided in four parts. Use the GROW-option to get the old style bars.
-



- \* Fixed a nasty bug in the AddTask() patch. A Commodore debugging tool TNT didn't work if it was run before Executive. This bug might have caused crashes with other programs too.
  - \* Executive now has a nice logo, you'll see it in the server and ExecutivePrefs icons. The logo is available in different formats, in the Executive WWW-page.
  - \* A sanity check is now done to the task names. If a non-printable character is found, it's converted to '?'.  
\* Meter can now display a graph of a value, for example CPU usage. The new options GRAPHHEIGHT, GRAPHS, GRAPHCOLOR and GRIDCOLOR are used to control the graphs.
  - \* Added some preliminary MUI V3 support:
    - A menu item "MUI Preferences..." which opens the MUI preferences editor.
    - Increased the size of some string gadgets, MUI V3 seems to make them a bit too small.
    - ALoad\_MUI can use the Levelmeter-class in MUI V3 to display a nice car speedometer like meter of current CPU usage. This looks really neat, and is very usable too. The meter is currently updated four times a second.
  - \* Miscellaneous internal changes and some polishing here and there.
  - \* The new Extra-drawer contains the full-size Executive logo and a screenshot of the Meter-client.
  - \* Fixed the Register Arexx-script.
  - \* A new option NOTITLEBAR in ALoad, ALoad3D, Meter and Top disables the window titlebar.
  - \* Updated the Portugues and Nederlands catalogs.
  - \* Added German catalog and Install/UnInstall translation.
  - \* Added French catalog and Install/UnInstall translation.
  - \* Some SysInfo.library modifications:  
(Thanks to Bernhard Möllemann <zza@rz.uni-karlsruhe.de>)
    - Better names for some structs:

|                   |    |                     |
|-------------------|----|---------------------|
| sysinfo           | -> | SysInfo             |
| loadaverage       | -> | SI_LoadAverage      |
| loadaverage_fixed | -> | SI_LoadAverageFixed |
| SysInfo_notify    | -> | SI_Notify           |
| cpu_usage         | -> | SI_CpuUsage         |
| task_cpu_usage    | -> | SI_TaskCpuUsage     |
    - SysInfo was sometimes spelled as sysinfo or Sysinfo. It's now always SysInfo.
    - sysinfo.fd is now SysInfo\_lib.fd
    - Changed the compiler dependant types:

|       |    |      |
|-------|----|------|
| int   | -> | LONG |
| short | -> | WORD |
-

- long -> LONG
- Removed LOADAVG\_FLOAT.
- Replaced the 'use\_messages' parameter in AddNotify() call with 'flags'.
- Some internal changes to support Executive V1.30

## 1.34 Acct

ACCT

~~~~~

Accounting daemon.

OPTIONS
DESCRIPTION

Acct writes

 accounting
 records to disk. Server creates an accounting
record every time a program started from CLI or Workbench terminates.

To reduce disk access Acct purges records to disk only after certain amount of them has been created or when the oldest record has been in memory for a user specified time. Acct doesn't use any CPU time when it's waiting for records.

The easiest way to start Acct is to copy it to your WBStartup-drawer. The Acct-client's icon should include a DONOTWAIT-tooltype, which tells Workbench not to wait for Acct to exit before proceeding.

It's not necessary to put Acct to WBStartup-drawer, you can still start it anytime you want to, from Workbench or from CLI. If you start it from CLI, you have to run it:

```
run <NIL: >NIL: Acct
```

The default accounting file is 's:acct', but you may change this. Use the ACCTFILE-option to do this or set the environmental variable ACCTFILE to contain the file name.

If the disk where accounting records are being written becomes full, accounting records might be lost. Accounting won't be disabled so you can make more room to the disk and Acct will continue normally. On heavily loaded systems 100-200KB of accounting records can be created each day, but a normal amount is 5-20KB.

To turn accounting off, send a break-signal to the accounting daemon or use the command:

```
Acct off
```

If you're using Workbench, start the Acct-client again and it will ask you if you want to turn accounting off.

If Acct notices an unprintable character in the command name, it will be changed to '?' to prevent cluttered display.

Acct works by patching the dos.library RunCommand() function. If you run another program that patches this function, it's not possible to quit Acct before the patch is removed. This is not very likely so you don't have to worry about it. A more common problem when quitting Acct is that some commands are still being executed and they have called the RunCommand() function, so it's not possible to safely remove the patch. If the patch can't be removed, Acct will notify you about this. Accounting will be disabled and Acct stays in memory waiting until the patch can be removed. You don't have to worry about this, you can safely start Acct again if necessary.

EXAMPLES

Start accounting and output all records to screen instead of writing them to disk. Use CTRL-C to stop.

```
Acct STDOUT
```

1.35 Acct options

ACCT OPTIONS

~~~~~

#### ACCTFILE

```
Template: F=ACCTFILE
Tooltype: FILE
Default:  s:acct
```

The accounting file name.

#### OFF

```
Template: OFF/S
Tooltype: OFF
```

Turn accounting off.

#### SECONDS

```
Template: S=SECONDS/N/K
Tooltype: SECONDS
Default:  30
```

Number of seconds to wait before writing records to disk. You can set this to 0, but then the disk must be accessed every time a command terminates and this may not be a good idea if you don't have a disk cache.

---

## RECORDS

Template: R=RECORDS/N/K  
Tooltype: RECORDS  
Default: 10

The maximum number of accounting records in memory at once. When this limit is reached, records will be written to disk.

## STDOUT

Template: STDOUT/S  
Tooltype: STDOUT

Instead of writing records to disk, display them on screen. This will also turn the COOKED-option on. See

Lastcomm  
for the output format.

## COOKED

Template: C=COOKED/S  
Tooltype: COOKED

Write accounting records in human-readable form, not in binary. If you write records to disk in this format, you can't use

Lastcomm  
or  
Sa  
to manipulate them. See  
Lastcomm  
for the output format.

## PURGE

Template: PURGE/S  
Tooltype: PURGE

Force Acct to write all records currently in memory to disk. Do this before resetting your system.

## HELP

Template: HELP/S  
Tooltype: HELP

Display this text.

## VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

---

## 1.36 ALoad

### ALOAD

~~~~~

ALoad displays a periodically updating histogram of system load average or current CPU usage.

OPTIONS DESCRIPTION

Two versions of ALoad are provided, a GadTools-version for all Amigas and one which supports MUI. They are identical except where otherwise said.

ALoad opens a small window and displays a periodically updating histogram of the system load average over last minute, or current CPU usage.

When load average reaches 1.0, the window is divided in two with a line through the middle. This line represents load average 1.0, the top of the window represents load average 1.99999... When load average reaches 2.0, the window will be divided in three sections.

The graph is scrolled to left when it reaches the right border. You can resize the window without losing the old histogram, which will be redrawn to the new window.

If you have MUI V3 you can use the GAUGE-option to display a small car speedometer like meter of current CPU usage. The meter is updated four times a second. A small damping factor is used to slow down the meter's needle so that it doesn't jump all over the place. Because the MUI V3 Levermeter-class is very small, no label is displayed for the window. There's no size gadget either because the class is not scalable. The Scroll, Update, Scale and Type menus are disabled. You'll probably want to set the window border width to 0 using MUI preferences.

ALoad can optionally display a clock in the window's titlebar.

There are several menu items available:

Project->New will redraw the histogram.

Settings->Update sets the update rate in seconds.

Settings->Scroll sets the amount that the histogram is scrolled when it reaches the right edge. 1/2 means that the histogram will be scrolled half the width of the current window.

Settings->Scale sets the MINIMUM scale of the histogram, i.e. the minimum number of divisions in the scale.

Settings->Type is used to choose what value is displayed,

system load average or current CPU usage.

Settings->Clock toggles the titlebar clock on/off.

Settings->Stay front makes the window stay in front of all other windows.

Settings->MUI Preferences... opens then MUI preferences editor. This requires MUI 3.0.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

EXAMPLES

Load average:

ALoad

Current CPU usage:

ALoad CURRENT

Scroll 10 pixels, update every 30 seconds, don't display a size-gadget:

ALoad SCROLL=10 UPDATE=30 NOSIZEGADGET

1.37 ALoad options

ALOAD OPTIONS

SCROLL

Template: SCR=SCROLL/N/K
 Tooltype: SCROLL
 Default: 1

The number of pixels to shift the graph to the left when the graph reaches the right edge of the window. If you specify value 0 here, it means that the histogram will be scrolled half the width of the current window.

UPDATE

Template: UD=UPDATE/N/K
 Tooltype: UPDATE
 Default: 1

The update interval in seconds.

SCALE

Template: SCA=SCALE/N/K
Tooltype: SCALE
Default: 1

The minimum number of tick marks in the histogram, where one division represents one load average point. If the load goes above this number, ALoad will create more divisions, but it will never use fewer than this number.

CURRENT

Template: C=CURRENT/S
Tooltype: CURRENT
Default: display system load average

Display current CPU usage instead of system load average.

BGCOLOR

Template: BGC=BGCOLOR/N/K
Tooltype: BGOLOR

Background color. (0-255)

COLOR

Template: CO=COLOR/N/K
Tooltype: COLOR

Graph color. (0-255)

SCALECOLOR

Template: SCC=SCALECOLOR/N/K
Tooltype: SCALECOLOR

Scale line color. (0-255)

CLOCK

Template: CLOCK/S
Tooltype: CLOCK

Display a clock in window titlebar.

LABEL

Template: LB=LABEL/K
Tooltype: LABEL

The text in the window's titlebar. You can put here whatever you like.

NOTITLEBAR

Template: NTB=NOTITLEBAR/S

Tooltype: NOTITLEBAR

Don't display a titlebar for the window. This option is not available in the MUI version of ALoad.

NOSIZEGADGET

Template: NSG=NOSIZEGADGET/S

Tooltype: NOSIZEGADGET

Don't display a size-gadget for the window.

STAYFRONT

Template: SF=STAYFRONT/S

Tooltype: STAYFRONT

Keep the window in front of all other windows.

LEFT

Template: LEFT/N/K

Tooltype: LEFT

Offset of the left edge of the window relative to screen. This option is not implemented in the MUI-version of ALoad.

TOP

Template: TOP/N/K

Tooltype: TOP

Offset of the top edge of the window relative to screen. This option is not implemented in the MUI-version of ALoad.

WIDTH

Template: WIDTH/N/K

Tooltype: WIDTH

Default: Calculated from display width

Width of the ALoad window. This option is not implemented in the MUI-version of ALoad.

HEIGHT

Template: HEIGHT/N/K

Tooltype: HEIGHT

Default: Calculated from display height

Height of the ALoad window. This option is not implemented in the MUI-version of ALoad.

PUBSCREEN

Template: PS=PUBSCREEN/K

Tooltype: PUBSCREEN

Default: default public screen

Name of the public screen where the window is to be opened.
This option is not implemented in the MUI-version of ALoad.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

1.38 ALoad3D

ALOAD3D

~~~~~

ALoad3D displays a periodically updating 3D bar-chart of system load averages.

#### OPTIONS DESCRIPTION

ALoad3D opens a small window and displays a periodically updating three dimensional bar-chart of the system load averages over last minute, last 5 minutes and last 15 minutes.

The maximum displayed load average is 5.0. If a load average is higher than this, then a small 'pyramid' is generated on top of the bar. Server recalculates load averages once a second.

There are several menu items available:

Settings->Delay between updates sets the delay in 1/50 seconds between display updates.

Settings->Speed sets the rotation speed (degrees).

Settings->Grow selects old style bars.

Settings->Motion->None stops the rotation.

Settings->Motion->Rotate rotates the bar-chart.

Settings->Motion->Wiggle makes the bar-chart rotate from left to right and then from right to left.

Settings->Stay front makes the window stay in front of all other

windows.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

You can rotate the bar-chart using left and right cursor keys.

ALoad3D requires a registered version of Executive.

#### EXAMPLES

ALoad3D

## 1.39 ALoad3D options

### ALOAD3D OPTIONS

#### DELAY

Template: D=DELAY/N/K  
Tooltype: DELAY  
Default: 3

Delay between display updates in 1/50 seconds. Maximum delay is 50 (one second). If the delay is 0, display will be constantly updated.

#### SPEED

Template: S=SPEED/N/K  
Tooltype: SPEED  
Default: 3

The rotation speed, maximum is 10.

#### GROW

Template: GROW/S  
Tooltype: GROW

Use old style bars.

#### NOMOTION

Template: NM=NOMOTION/S  
Tooltype: NOMOTION

Don't rotate the bar-chart.

#### WIGGLE

Template: WG=WIGGLE/S  
Tooltype: WIGGLE

Makes the bar-chart rotate from left to right and then from right to left.

#### ANGLE

Template: A=ANGLE/N/K  
Tooltype: ANGLE  
Default: 0

Default rotation angle.

#### LABEL

Template: LB=LABEL/K  
Tooltype: LABEL

The text in the window's titlebar. You can put here whatever you like.

#### NOTITLEBAR

Template: NTB=NOTITLEBAR/S  
Tooltype: NOTITLEBAR

Don't display a titlebar for the window.

#### NOSIZEGADGET

Template: NSG=NOSIZEGADGET/S  
Tooltype: NOSIZEGADGET

Don't display a size-gadget for the window.

#### STAYFRONT

Template: SF=STAYFRONT/S  
Tooltype: STAYFRONT

Keep the window in front of all other windows.

#### LEFT

Template: LEFT/N/K  
Tooltype: LEFT

Offset of the left edge of the window relative to screen.

#### TOP

Template: TOP/N/K  
Tooltype: TOP

Offset of the top edge of the window relative to screen.

---

## WIDTH

Template: WIDTH/N/K  
Tooltype: WIDTH  
Default: Calculated from display width

Width of the ALoad3D window.

## HEIGHT

Template: HEIGHT/N/K  
Tooltype: HEIGHT  
Default: Calculated from display height

Height of the ALoad3D window.

## PUBSCREEN

Template: PS=PUBSCREEN/K  
Tooltype: PUBSCREEN  
Default: default public screen

Name of the public screen where the window is to be opened.

## HELP

Template: HELP/S  
Tooltype: HELP

Display this text.

## VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.40 Commander

### COMMANDER

~~~~~

A general task-manager with GUI.

OPTIONS DESCRIPTION

Two versions of Commander are provided, a GadTools-version for all Amigas and one which supports MUI. They are identical except where otherwise said.

Commander opens a window showing all tasks currently in system. The

display looks like this:

NAME	PID	TYPE	PRI	RPRI	NICE
[CyberCron]	4	bcli	-70	0	0
console.device	208	task	5	5	0
DH0	247	proc	-63	1	-20
ramlib	209	proc	-70	0	0
...					

After the task name and its PID, the TYPE field shows if the task is a task, process, cli or background cli. PRI is the task's priority, RPRI is its real priority, the one that will be returned when the server quits. NICE is, obviously, task's nice-value.

If the task name is in square brackets, it's a CLI command.

BREAK

The break-button lets you send a break-signal to the select task. You can choose from four different signals, usually you'll use the C signal, which is used to break a task. D-signal is used with shell scripts and F-signal is used with commodity-programs. F-signal is currently undefined. Note that Commander only sends the break-signal to a task, it can't guarantee that the task will actually exit.

NICE

This lets you modify tasks nice-value. Higher values make the task 'nicer', i.e. it will be given less CPU time. Negative values will do the opposite.

SIGNAL

Each task can receive 32 different signals. If you don't know anything about signals, press the cancel-button and don't try to signal a task, it will probably just crash your system.

All the 32 signals are shown in the window. You can choose any number of signals you want to send. If a signal is shown in white, it means that the task is currently waiting for this signal. If the signal has been ghosted, it's not allocated and sending it has no effect.

REMOVE

Will remove the selected task from exec.library task-list by calling RemTask(). Task's resources won't be deallocated and a crash is very likely. Don't use this if you don't know exactly what you're doing.

PRIORITY

Set the priority of a task. If you change task's priority to catch range, scheduler will notice this and start scheduling the task.

UPDATE

Update the task list. The task list is not updated automatically, so it may include tasks that have long since terminated.

QUIT

Obviously, quits Commander.

The MUI version of Commander also displays a CPU usage of the selected task in a graph on top of the window. The default is to show recent CPU usage, but choose the Settings->MeterType->Current CPU usage menu item to show last second CPU usage.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

Settings->MUI Preferences... opens then MUI preferences editor. This requires MUI 3.0.

EXAMPLES

Commander

1.41 Commander options

COMMANDER OPTIONS

~~~~~

#### CURRENT

Template: C=CURRENT/S  
Tooltype: CURRENT

Instead of showing recent CPU usage, show current, last second CPU usage. This option is available only in the MUI-version of Commander.

#### ZIP

Template: ZIP/S  
Tooltype: ZIP

Open Commander's window so that only the titlebar is displayed. This option is available only in the GadTools-version of Commander.

#### LEFT

Template: LEFT/N/K  
Tooltype: LEFT

Offset of the left edge of the window relative to screen. This option is not implemented in the MUI-version of Commander.

#### TOP

Template: TOP/N/K

---

Tooltype: TOP

Offset of top edge of the window relative to the screen.  
This option is not implemented in the MUI-version of Commander.

HELP

Template: HELP/S  
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.42 Ctp

CTP

~~~~~

Change task's priority.

OPTIONS
DESCRIPTION

Ctp is similar to the standard Amiga utility ChangeTaskPri, but it has a lot more options.

Where ChangeTaskPri can only change the priority of a one task at a time, Ctp can change multiple tasks' priorities with one command. For example:

```
Ctp PRI=-1 PID=10,20 NAME=DH\* USER=root GROUP=staff PGRP=273
```

The following tasks will be affected by this command:

(NOTE: Depending on the shell you use, the \ may not be needed in front of the asterisk.)

- tasks with PID 10 and 20
- tasks with name starting with 'DH' (case insensitive)
- all tasks owned by the root user. (MultiUser only)
- all tasks owned by the staff group. (MultiUser only)
- all tasks that belong to process group 273.

All these tasks get the specified priority, -1.

If you don't specify any task, priority of the current task (usually a CLI process) will be changed.

There's no need to use the PRI- and PID-keywords, Ctp works without

them too:

```
Ctp -10 278
```

This will change the priority of the task with PID 278 to -10.

You can replace `c:ChangeTaskPri` with `Ctp`, they are compatible and `Ctp` doesn't need the server to run. If the server isn't running you can use only PIDs that correspond to

CLI numbers

.

When `MultiUser` is installed, only the super-user may raise task's priority. If you don't want to require root privileges for `Ctp`, set the U-bit using `MProtect`.

EXAMPLES

Change the priority of the current task to 0.

```
Ctp 0
```

Change the priority of 'RexxMaster' to 5.

```
Ctp 5 NAME=RexxMaster
```

1.43 Ctp options

CTP OPTIONS

~~~~~

#### PRIORITY

Template: PRI=PRIORITY/N/A

Tooltype: PRIORITY

New priority for the specified tasks.

#### PID

Template: PID=PROCESS

Tooltype: PID

Default: Current task

PIDs of the tasks whose priority is to be changed. You can list multiple PIDs here, separate them with a comma, for example 'PID=10,248,332'.

#### NAME

Template: N=NAME/K

Tooltype: NAME

Names of the tasks whose priority is to be changed. You can list multiple



names here, separate them with a comma, for example ``NAME=DH0,DH1,DH2,*.  
*.device'`. You can use `*` as a wildcard.

#### USER

Template: `U=USER/K`  
Tooltype: `USER`

Names of the users whose all tasks will get the new priority. You can list multiple user names here, separate them with a comma, for example ``USER=Frodo,Gandalf,Aragorn'`. You can also use UIDs. This option isn't available if MultiUser hasn't been installed.

#### GROUP

Template: `G=GROUP/K`  
Tooltype: `GROUP`

Names of the groups whose all tasks will get the new priority. You can list multiple group names here, separate them with a comma, for example ``NAME=staff,students'`. You can also use GIDs. This option isn't available if MultiUser hasn't been installed.

#### PGRP

Template: `R=PGRP/K`  
Tooltype: `PGRP`

PGRPs of the process groups whose priority is to be changed. All tasks in the process group will be affected. You can list multiple PGRPs here, separate them with a comma, for example ``PGRP=254,298,578'`.

#### HELP

Template: `HELP/S`  
Tooltype: `HELP`

Display this text.

#### VERSION

Template: `VERSION/S`  
Tooltype: `-`

Display version number.

## 1.44 ExecutivePrefs

### EXECUTIVEPREFS

~~~~~

Change server preferences.

OPTIONS
DESCRIPTION

Two versions of ExecutivePrefs are provided, a GadTools-version for all Amigas and one which supports MUI. They are identical except where otherwise said.

ExecutivePrefs searches the preferences file from these directories, in this order:

```
current directory
PROGDIR: (directory where the ExecutivePrefs binary is)
ENV:
S:
```

If the file can't be loaded, default settings are used.

The Save-button saves the preferences back to the file that was loaded, or to 'PROGDIR:Executive.prefs'-file if no preferences file was loaded. The changes won't take effect before the server is restarted.

The Save/Restart-button saves the preferences file and restarts the server. You'll lose information about window owners and task relationships if you restart the server.

ExecutivePrefs is divided into four sections:

```
Tasks
Options
Ranges
Scheduler
```

The Tasks-section lets you add a task to a list of watched tasks. A list of tasks currently being watched is displayed, it might look like this:

WHICH	NAME	TYPE	PRIORITY	NICE
TASK	Top	SCHEDULE		-15
CHILDTASKS	ASwarm	NOSCHEDULE	BELOW	
IGNORE	BuggyProgram*			
...				

The WHICH-field is TASK, CHILDTASKS, TASK+CHILDTASKS or IGNORE. TASK affects the named task only, CHILDTASKS doesn't affect the named task, but all its childevents. TASK+CHILDTASKS affects the named task and its childevents. If WHICH is IGNORE, then the named task will be totally ignored. The IGNORE-option is VERY rarely needed with tasks that aren't removed from the system in a legal way, i.e. calling exec.library RemTask() function.

NAME is the task name. Ps, Top and some other clients might display a task name in square brackets, which means that it's a CLI command. The brackets don't actually belong to the task name, so don't use the brackets here.

TASK, CHILDTASKS and TASK+CHILDTASKS require that the TYPE is also specified. TYPE-field is either SCHEDULE or NOSCHEDULE. SCHEDULE forces

a task to be scheduled, NOSCHEDULE tells the server NOT to schedule the task, even if its priority would be in the catch range.

PRIORITY is used only with NOSCHEDULE. If it's blank, nothing will be done to task's priority. If it's ABOVE, the priority will be kept above the catch range. BELOW will keep the priority below the catch range. If it's a number, then the affected task(s) will be given this priority.

NICE is used only with SCHEDULE, it lets you set a default nice-value for the task.

See the

server documentation
for more information.

IMPORTANT! For Executive to catch a task, you must start it from CLI with the Run-command, or start it from Workbench. Because if you just execute the command in a CLI, no new task is created and Executive won't notice the command.

The New-button creates a new entry and opens the edit-window, which can also be opened by double-clicking a task, or pressing the Edit-button.

The Copy-button makes a copy of the selected task and the Delete-button deletes the selected task.

The Edit-window has got buttons to change all the above values. First you can change the name of the task, and as said earlier, * is allowed as a wildcard. By pressing the popup-button in the MUI version of ExecutivePrefs or the Name-button in the GadTools version, you can choose a name from a list of tasks currently in running.

The * MAGIC WAND * -button opens a list of preset entries for certain programs. When you select a task from this list, the settings will be adjusted and you can just press the Ok-button to confirm. Sometimes you might need to give a name to the task, if it's not specified in the Magic Wand.

I have included several common screenblankers, multi-function commodities and communication programs to the magic wand -list, if you know other programs that should be there, please let me know.

You'll also find a Magic Wand for ShapeShifter Mac emulator, which doesn't seem to work when it's subtasks' priorities are changed.

The explanation window reflects the state of the buttons below it. It will explain what will happen to the task. Try different settings and see how it changes.

The Options-section is used to set a couple of miscellaneous options. 'No focus on active window' disables focus on active window. 'Schedule Workbench' will

schedule Workbench
also. You can also change the

preferred

```

        timer
        from here. 'Don't fix OpenWorkbench bug' tells the
server not to fix a
        bug
        in AmigaOS 3.x.

```

The Ranges-section lets you adjust the catch and dynamic ranges

ExecutivePrefs makes sure that you don't specify illegal values here. It permits you to set a high maximum priority for both ranges, but try to avoid setting the catch range maximum value over 4.

The Scheduler-section lets you choose a scheduler

Remember that the changes you make don't affect the currently running server, you must restart it.

The Settings->MUI Preferences... opens then MUI preferences editor. This requires MUI 3.0.

EXAMPLES

```
ExecutivePrefs
```

1.45 ExecutivePrefs options

EXECUTIVEPREFS OPTIONS

FROM

```
Template: FROM
Tooltype: FROM
```

Name and path of the preferences file if it isn't in any of the directories where ExecutivePrefs looks for it first.

LEFT

```
Template: LEFT/N/K
Tooltype: LEFT
```

Offset of the left edge of the window relative to screen. This option is not implemented in the MUI-version of ExecutivePrefs.

TOP

```
Template: TOP/N/K
Tooltype: TOP
```

Offset of the top edge of the window relative to screen. This option is not implemented in the MUI-version of ExecutivePrefs.

WRITEDEFAULTS

Template: WRITEDEFAULTS/K
Tooltype: -

The installation script uses this option to write a default preferences file.

UPDATEPREFS

Template: UPDATEPREFS/S
Tooltype: -

The installation script uses this option to update existing preferences file.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

1.46 Kill

KILL

~~~~~  
Send a break-signal to task.

### OPTIONS DESCRIPTION

Kill is similar to the standard Amiga utility Break, but it has a lot more options. Kill sends a break signal to specified tasks but it's not guaranteed that these tasks will react to the signal.

Where Break can only send break-signal to one task at a time, Kill can handle several tasks with one command. For example:

```
Kill PID=10,20 NAME=DH\* USER=root GROUP=staff PGRP=273
```

The following tasks will be affected by this command:

---

(NOTE: Depending on the shell you use, the \ may not be needed in front of the asterisk.)

- tasks with PID 10 and 20
- tasks with name starting with 'DH' (case insensitive).
- all tasks owned by the root user. (MultiUser only)
- all tasks owned by the staff group. (MultiUser only)
- all tasks that belong to process group 273.

There's no need to use the PID-keyword, Kill can be invoked without it too:

```
Kill 10
```

This will send break-signal to task with PID 10.

You can replace c:Break with Kill, they are compatible and Kill doesn't need the server to run. If the server isn't running you can use only PIDs that correspond to

CLI numbers

.

You can't kill the server with Kill, because Kill connects to the server and the server can't exit if it gets a break-signal because Kill has connected to it.

The PGRP-option is really useful when you, for example, want to kill a GNU C compilation. 'Pstree SHOWPIDS' outputs:

```
`---[Make] (9)
  +---CON (236)
    `---[sh] (1)
      `---[cpp] (10)
```

You can send a break-signal to all these tasks with a single command:

```
Kill PGRP=9
```

If MultiUser has been installed, then root privileges are needed to kill other users' tasks. If you don't want to require root privileges for Kill, set the U-bit using MProtect.

## EXAMPLES

Sorry, it's not possible to give any good examples of Kill.

## 1.47 Kill options

### KILL OPTIONS

~~~~~  
PID

Template: PID=PROCESS
Tooltype: PID

PIDs of the tasks that will be killed. You can list multiple PIDs here, separate them with a comma, for example 'PID=10,248,332'.

NAME

Template: N=NAME/K
Tooltype: NAME

Names of the tasks that will be killed. You can list multiple names here, separate them with a comma, for example 'NAME=DH0,DH1,DH2,Shell*'. You can use * as a wildcard.

USER

Template: U=USER/K
Tooltype: USER

Names of the users whose all tasks will be killed. You can list multiple user names here, separate them with a comma, for example 'USER=Frodo,Pippin,Gandalf'. You can also use UIDs. This option isn't available if MultiUser hasn't been installed.

GROUP

Template: G=GROUP/K
Tooltype: GROUP

Names of the groups whose all tasks will be killed. You can list multiple group names here, separate them with a comma, for example 'NAME=staff,students'. You can also use GIDs. This option isn't available if MultiUser hasn't been installed.

PGRP

Template: R=PGRP/K
Tooltype: PGRP

PGRPs of the process groups whose all tasks will be killed. You can list multiple PGRPs here, separate them with a comma, for example 'PGRP=254,298,578'.

ALL

Template: ALL/S
Tooltype: ALL

Send all signals (CDEF) to the task(s).

C

Template: C/S
Tooltype: C

Send the C-signal to the task(s). This is the default and most

commonly used.

D

Template: D/S
Tooltype: D

Send the D-signal to the task(s).

E

Template: E/S
Tooltype: E

Send the E-signal to the task(s).

F

Template: F/S
Tooltype: F

Send the F-signal to the task(s).

REMTASK

Template: REMTASK/S
Tooltype: REMTASK

This will REMOVE the specified task(s) from Exec task-lists with RemTask() call. This is EXTREMELY DANGEROUS and you can easily crash your system if you don't know exactly what you are doing.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

1.48 Lastcomm

LASTCOMM

~~~~~

Show last commands executed in reverse order.



OPTIONS  
DESCRIPTION

Lastcomm gives information on previously executed commands. With no arguments, Lastcomm displays information about all the commands recorded during the current accounting file's lifetime. The most recently terminated command will be shown first.

Lastcomm will read from the default accounting file 's:acct', but you can use the ACCTFILE-option or the environmental variable ACCTFILE to specify another name for the accounting file.

The output looks like this:

```
login          -          --H    0.06s  14-Jun-95 00:04:17
lzx           root        S-H    0.55s  14-Jun-95 00:04:32
lha           -          --H   16.04s  14-Jun-95 00:04:07
gzip          -          --H    3.36s  14-Jun-95 00:03:53
csh           -          --H    0.12s  14-Jun-95 00:03:29
endcli        -          --H    0.00s  14-Jun-95 00:03:29
date          -          ---    0.01s  14-Jun-95 00:03:28
stack         -          ---    0.01s  14-Jun-95 00:03:28
RX            -          --H    0.03s  14-Jun-95 00:03:05
Sa            -          --H    0.84s  14-Jun-95 00:03:04
```

For each command, the following are displayed:

- command name
- name of the user who executed the command (requires MultiUser)  
(If MultiUser hasn't been installed, the user name will be -)
- flags:
  - S -> command was executed by super-user
  - W -> command was run from Workbench
  - H -> command was scheduled
- amount of CPU time used by the command (in seconds)
- time the command terminated

Lastcomm doesn't require that the server is running.

Lastcomm asks the Acct-daemon to write all unpurged records to disk so they will be included in the output.

CTRL-C will stop the output.

## EXAMPLES

```
Lastcomm
```

## 1.49 Lastcomm options

### LASTCOMM OPTIONS

~~~~~

ACCTFILE

Template: F=ACCTFILE
Tooltype: ACCTFILE
Default: s:acct

The accounting file name.

PROCS

Template: C=COMMANDS/N/K
Tooltype: COMMANDS
Default: all

How many commands to display.

NAME

Template: N=NAME/K
Tooltype: NAME

Only commands matching this name should be displayed. You can use * as a wildcard.

USER

Template: U=USER/K
Tooltype: USER

Display commands executed by this user (name or UID). This option isn't available if MultiUser hasn't been installed.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

1.50 Meter

METER

~~~~~

Display system information.

OPTIONS

---

## DESCRIPTION

Meter displays free memory, load averages, CPU usage, uptime, free disk space and lots of other things in a small window. Meter will display each piece of information on one line. A horizontal bar will be displayed if possible. Information in the window is updated by default every second.

Bars indicate a percentage of a current value from a maximum value. For example, currently free memory from total available memory. For some values, it's not possible to display a bar, because there's no maximum value. Maximum value here means for example total available memory.

It's also possible to display a small graph for values that have a maximum value. The graph is drawn on top of the bar and the value name. The default graph height is 2x font height.

To make the graphs more readable, a few scale lines will be drawn. One solid line on the bottom and a dotted line in the middle and. For load average values this is different, scale lines are drawn for each load average level, like in ALoad.

You can choose from the Show-menu what information should be displayed. The window will be resized if necessary. There are a few presets in the Settings->Presets menu:

|              |                              |
|--------------|------------------------------|
| Default      | Show default values          |
| User defined | Show the user defined values |
| Memory       | Show memory usage            |
| CPU usage    | Show CPU usage               |
| VMM stats    | Show VMM statistics          |

Using the SHOW-option you can specify what information should be displayed. SHOW-option takes a list of keywords separated by a comma. A 'G' indicates that a graph can be drawn for this value:

|                 |                                                 |
|-----------------|-------------------------------------------------|
| clock           | Display a clock                                 |
| chip            | G Free CHIP memory                              |
| fast            | G Free FAST memory                              |
| virtual         | G Virtual memory (GigaMem or VMM)               |
| total           | G Total free memory                             |
| largest         | G Largest free block                            |
| load1           | G Load average over the last minute             |
| load5           | G Load average over the last 5 minutes          |
| load15          | G Load average over the last 15 minutes         |
| cpuusage        | G CPU usage                                     |
| recentcpu       | G Recent CPU (last minute)                      |
| unpurged        | Number of unpurged accounting records           |
| acctrecs        | Total number of accounting records created      |
| addedtasks      | Number of added tasks                           |
| terminatedtasks | Number of terminated tasks                      |
| tasksrunning    | Tasks currently running or ready to run         |
| currentcsw      | Number of context switches during last second   |
| currentvocsw    | Voluntary context switches during last second   |
| currentivcsw    | Involuntary context switches during last second |
| csw             | Total number of context switches                |

|              |                                  |       |
|--------------|----------------------------------|-------|
| vocsw        | Voluntary context switches       |       |
| ivcsw        | Involuntary context switches     |       |
| users        | Users logged in (MultiUser only) |       |
| uptime       | Uptime                           |       |
| pagefaults   | Pagefaults                       | (VMM) |
| pagesread    | Pages read                       | (VMM) |
| pageswritten | Pages written                    | (VMM) |
| pageframes   | Pageframes                       | (VMM) |
| pagesused    | Pages on dev                     | (VMM) |

You can also use device names like DF0:, DH0:, RAM: etc. as keywords. The amount of free disk space will be displayed for these devices. If there's no disk in drive or the disk is unreadable, a "-" is displayed here.

To display total free memory, CPU usage, uptime and free space on drive DH0: use this command:

```
Meter SHOW=total,cpuusage,uptime,DH0:
```

The order of the keywords is insignificant, they will always be in the same order in the window.

I suggest you use some small font so you can display more information in the window. You can use the BARSPACE-option to control the number of 1-pixel lines Meter inserts between values and the BAREXTRA-option to specify extra height for bars.

Use the options BGCOLOR, BARCOLOR, TEXTCOLOR, NUMBERCOLOR and GRAPHCOLOR to change background, bar, text, number and graph colors, respectively.

If you don't want to display any bars, labels or numbers, use the three menu items in the Settings menu:

```
Settings->Show labels  disable/enable labels
Settings->Show numbers "          numbers
Settings->Show bars   "          bars
Settings->Show graphs  "          graphs
```

The menu item Project->New will redraw the display. Project->Flush memory flushes all unused libraries from memory. Settings->Stay front keeps Meter's window in front of all other windows.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

Meter requires a registered version of Executive.

## EXAMPLES

Meter

CPU usage as a graph (+ clock):

Meter SHOW=clock,cpuusage GRAPHS GRAPHHEIGHT=150

Use a different font:

Meter FONT=courier FONTSIZE=13

Output a list of keywords:

Meter LIST

## 1.51 Meter options

METER OPTIONS

~~~~~  
SHOW

Template: S=SHOW
Tooltype: SHOW
Default: CLOCK,CHIP,FAST,TOTAL,LARGEST,LOAD1,CPUUSAGE,RECENTCPU,
CURRENTCSW,UPTIME

What information Meter should display. Separate keywords with a comma.
Order of the keywords is insignificant.

UPDATE

Template: UD=UPDATE/N/K
Tooltype: UPDATE
Default: 1

The update interval in seconds.

GRAPHHEIGHT

Template: GH=GRAPHHEIGHT/N/K
Tooltype: GRAPHHEIGHT
Default: 2x font height

Height of the graphs in pixels.

BARSPACE

Template: BS=BARSPACE/N/K
Tooltype: BARSPACE
Default: 1

How many empty 1-pixel lines between bars.

BAREXTRA

Template: BE=BAREXTRA/N/K
Tooltype: BAREXTRA

Default: 0

Bar height is by default the same as the font height, but you can use this option to make the bars taller. Specifying 1 makes the bar two pixels taller, one extra line above the bar and one below.

BGCOLOR

Template: BGC=BGCOLOR/N/K
Tooltype: BGCOLOR

Background color. (0-255)

BARCOLOR

Template: BARC=BARCOLOR/N/K
Tooltype: BARCOLOR

Bar color. (0-255)

TEXTCOLOR

Template: TXTC=TEXTCOLOR/N/K
Tooltype: TEXTCOLOR

Text color. (0-255)

NUMBERCOLOR

Template: NBC=NUMBERCOLOR/N/K
Tooltype: NUMBERCOLOR

Number color. (0-255)

GRAPHCOLOR

Template: GRC=GRAPHCOLOR/N/K
Tooltype: GRAPHCOLOR

Graph color. (0-255)

GRIDCOLOR

Template: GRD=GRIDCOLOR/N/K
Tooltype: GRIDCOLOR

Color of the graph scale lines. (0-255)

NOLABELS

Template: NOLABELS/S
Tooltype: NOLABELS

Don't display labels.

NONUMBERS

Template: NONNUMBERS/S
Tooltype: NONNUMBERS

Don't display numbers.

NOBARS

Template: NOBARS/S
Tooltype: NOBARS

Don't display bars.

GRAPHS

Template: GRAPHS/S
Tooltype: GRAPHS

Display graphs.

SCROLL

Template: SCROLL/N/K
Tooltype: SCROLL
Default: 1

Number of pixels the graphs are scrolled each time.

MEMORY

Template: MEM=MEMORY/S
Tooltype: MEMORY

Display free memory.

CPUUSAGE

Template: CPU=CPUUSAGE/S
Tooltype: CPUUSAGE

Display CPU usage.

VMMSTATS

Template: VMM=VMMSTATS/S
Tooltype: VMMSTATS

Display VMM statistics.

LIST

Template: LIST/S
Tooltype: LIST

List all keywords.

LABEL

Template: LB=LABEL/K
Tooltype: LABEL

The text in the window's titlebar. You can put here whatever you like.

NOTITLEBAR

Template: NTB=NOTITLEBAR/S
Tooltype: NOTITLEBAR

Don't display a titlebar for the window.

FONT

Template: F=FONT/K
Tooltype: FONT
Default: Screen font

Font to be used in Meter's window. You might want to use a small font, so you can display more information and make the window smaller.

FONTSIZE

Template: FS=FONTSIZE/N/K
Tooltype: FONTSIZE

If you specify a font, you should also specify the font size.

ZIP

Template: ZIP/S
Tooltype: ZIP

Open Meter's window so that only the titlebar is displayed.

STAYFRONT

Template: SF=STAYFRONT/S
Tooltype: STAYFRONT

Keep the window in front of all other windows.

PUBSCREEN

Template: PS=PUBSCREEN/K
Tooltype: PUBSCREEN
Default: default public screen

Name of the public screen where the window is to be opened.

LEFT

Template: LEFT/N/K
Tooltype: LEFT

Offset of the left edge of the window relative to screen.

TOP

Template: TOP/N/K
 Tooltype: TOP

Offset of the top edge of the window relative to screen.

HELP

Template: HELP/S
 Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
 Tooltype: -

Display version number.

1.52 Nice

NICE

~~~~~

Execute a command with higher nice-value.

### OPTIONS DESCRIPTION

Nice executes a command with a higher nice-value than the current task has. If no increment is given, 10 is assumed.

Arguments can be given for the command as usual. If you're using a custom shell like csh or sksh, it's possible to run commands under that shell. Several clients use this feature, read about it from [here](#)

.

There are two ways to give the nice-value increment:

With the NICE-keyword:

```
Nice NICE=10 <command>
```

Or by preceding the value with a '--':

```
Nice -10 <command>    =>  Nice NICE=10 <command>
Nice --10 <command>   =>  Nice NICE=-10 <command>
```

If the server isn't running, Nice will still run the command, but the nice-value is ignored.

Nice can also be used from Workbench, see  
Genies  
for more information.

When MultiUser is installed, only the super-user may use negative nice-value increments. If you don't want to require root privileges for Nice, set the U-bit using MProtect.

#### EXAMPLES

Execute dir with nice-value of this task + 10.

```
Nice dir
```

Execute dir with nice-value of this task + (-20).

```
Nice NICE=-20 dir
```

## 1.53 Nice options

### NICE OPTIONS

#### NICE

```
Template: N=NICE/N/K
Tooltype: NICE
Default: 1
```

Run the command with this task's nice-value plus this increment.

#### SHELL

```
Template: S=SHELL/K
Tooltype: SHELL
Default: Run command under standard Amiga shell
```

If you're using a custom shell, you can specify here a  
template  
that is  
used when starting a command under that shell.

#### HELP

```
Template: HELP/S
Tooltype: HELP
```

Display this text.

#### VERSION

```
Template: VERSION/S
```

Tooltype: -

Display version number.

#### COMMAND

Template: C=COMMAND/F

Tooltype: COMMAND

The command to be executed and its arguments.

## 1.54 Ps

PSTREE

~~~~~

Display information about tasks.

OPTIONS
DESCRIPTION

Ps displays information about all or selected tasks in system. The output is sorted by task PIDs.

The Ps's output is controlled with keywords. Each keyword displays a piece of information about a task, for example:

PID	task's PID
NAME	task name
TIME	total used cputime
PRI	task's priority

And so on, over hundred keywords are currently known. Use the LIST-keyword to get a list of all available keywords.

The default output consists of these keywords:

PID,PRI,RPRI,NICE,TYPE,STATE,TIME,IDLE,CPU%,NAME

If you would like to display only task's pid, priority and name, you would use this command:

Ps FORMAT=PID,PRI,NAME

Or just:

Ps PID,PRI,NAME

See the explanation of
all available keywords

.

If you want to align the header of a keyword to left, use < before the keyword and > if you want to align it to right. Some keywords are variable width and you can specify their width with /<width> after the keyword. For example, this will change the width of the NAME-field to 50 characters:

```
Ps PID,PRI,NAME/50
```

These options specify a built-in format string:

EVERYTHING	Output every keyword
CPU	Output CPU usage values
STACK	Output stack usage statistics
TASK	Output Exec's task-structure
PROCESS	Output DOS-process structure
CLI	Output CLI-process structure
BABY	Output Executive internal task structure

For example,

```
Ps CPU
```

will display several CPU usage values.

You can sort the list by any of these keywords, or even using multiple keywords. For example, if you want to sort by priority, use this command:

```
Ps SORT=PRI
```

If you want that tasks with the same priority are sorted by PID, use this command:

```
Ps SORT=PID,PRI
```

To display the 15 tasks that have been idle for shortest time, you need to sort in reverse. This is accomplished with a - in front of the keyword name.

```
Ps SORT=-IDLE NPROCS=15
```

Several options are available for selecting only a part of all tasks for output. For example, SCHEDULED-option displays only scheduled tasks, WAIT-option displays only waiting tasks and ACTIVE-option displays tasks that have recently used CPU time. See the `options` -section for other similar options.

It's possible to combine keywords like ACTIVE and SCHEDULED, this would display all scheduled tasks that have recently used CPU time:

```
Ps ACTIVE SCHEDULED
```

If you have installed MultiUser and have logged in as some else than root, Ps will display only those tasks that belong to you. To override this, use the option ALL.

CTRL-C will stop the output.

EXAMPLES

Ps

Show all devices:

Ps NAME=*.device

Show stack usage:

Ps FORMAT=PID,PRI,NICE,STACK,STACKUSE,STACK%,TIME,CPU%,NAME

1.55 Ps options

PS OPTIONS

~~~~~

#### FORMAT

Template: F=FORMAT

Tooltype: FORMAT

Default: PID,PRI,RPRI,NICE,TYPE,STATE,TIME,IDLE,CPU%,NAME

The format string. Separate keywords with a comma. Use < before the keyword to align the header to left and > to right. Use /<width> after the keyword to specify width for variable width keywords, e.g. NAME.

This will align the NAME-header to right and the width of this field will be 40 characters:

Ps FORMAT=>NAME/40

#### SORT

Template: S=SORT/K

Tooltype: SORT

Default: PID

The sort order. Separate keywords with a comma. Use - before the keyword to sort in reverse with this keyword.

It's sometimes useful to specify multiple keywords here, for example, if you want to sort tasks by CPU%, then tasks with 0 CPU% will be in random order. If you first sort with NAME, these tasks will be sorted by their name:

Ps SORT=NAME,CPU%

Try this and

Ps SORT=CPU%

compare the results!

#### LIST

Template: LIST/S  
Tooltype: LIST

List all keywords. This will display the keyword name, its header, alignment, width (fixed or variable), default width and minimum width (if variable width keyword).

#### NOHEADER

Template: NH=NOHEADER/S  
Tooltype: NOHEADER

Don't display the header.

#### PADHEX

Template: PH=PADHEX/S  
Tooltype: PADHEX

Hex numbers will be padded with 0:s, for example, fdc920 becomes 00fdc920.

#### PLAIN

Template: PLAIN/S  
Tooltype: PLAIN

Don't use ANSI codes in output.

#### DELAY

Template: DELAY/N/K  
Tooltype: DELAY  
Default: 0

Delay specified number of seconds before outputting the task list. I added this option so I could see if a screenblanker creates a task when it blanks the screen.

#### INVERSE

Template: IV=INVERSE/S  
Tooltype: INVERSE

Inverse task selection. For example:

```
Ps SCHEDULED INVERSE
```

output tasks that are NOT scheduled.

#### ACTIVE

---

Template: ACTIVE/S  
Tooltype: ACTIVE

Output tasks with non-zero CPU%.

#### IDLE

Template: IDLE/S  
Tooltype: IDLE

Output tasks with zero CPU%.

#### SCHEDULED

Template: SCHED=SCHEDULED/S  
Tooltype: SCHEDULED

Output scheduled tasks.

#### RUN

Template: RUN/S  
Tooltype: RUN

Output tasks that are running at the moment. If you don't have a multiprocessor Amiga, this will output only the current task. ;)

#### READY

Template: READY/S  
Tooltype: READY

Output tasks that are ready, but not running.

#### WAIT

Template: WAIT/S  
Tooltype: WAIT

Output tasks that are waiting.

#### ALL

Template: ALL/S  
Tooltype: ALL

If you have installed MultiUser and you have logged in as some else than root, Ps will display only those tasks that belong to you. Using this option you can override this and display all tasks. This option isn't available if MultiUser hasn't been installed.

#### OWN

Template: OWN/S  
Tooltype: OWN

If you have installed MultiUser, this option will display only those

---

tasks that belong to you. This is the default if you have logged in as some else than root. This option isn't available if MultiUser hasn't been installed.

#### PID

Template: P=PID/K  
Tooltype: PID

Display tasks with this PID. You can list multiple PIDs here, separate them with a comma.

#### USER

Template: U=USER/K  
Tooltype: USER

Display tasks that belong to these users, separate user names with a comma. This option isn't available if MultiUser hasn't been installed.

#### GROUP

Template: G=GROUP/K  
Tooltype: GROUP

Display tasks that belong to these groups, separate user names with a comma. This option isn't available if MultiUser hasn't been installed.

#### NAME

Template: N=NAME/K  
Tooltype: NAME

Display tasks that match the given name. You can use \* as a wildcard.

#### PGRP

Template: R=PGRP/K  
Tooltype: PGRP

Display tasks in this process group. You can list multiple pgrps here, separate them with a comma.

#### PRIGT

Template: PRIGT/N/K  
Tooltype: PRIGT

Display tasks with greater priority than the given value.

#### PRILT

Template: PRILT/N/K  
Tooltype: PRILT

Display tasks with lower priority than the given value.

---



## PRIEQ

Template: PRIEQ/N/K  
Tooltype: PRIEQ

Display tasks with the same priority as the given value.

## NPROCS

Template: NPROCS/N/K  
Tooltype: NPROCS

Maximum number of tasks to display.

## EVERYTHING

Template: EVERYTHING/S  
Tooltype: EVERYTHING

This will display information using ALL keywords. You should redirect the output to a file, because it will be hundreds of characters wide.

## BABY

Template: BABY/S  
Tooltype: BABY

Display the Executive internal task records, probably not very interesting, I use this for debugging.

## TASK

Template: TASK/S  
Tooltype: TASK

Display the information structure associated with each task.

## PROCESS

Template: PROCESS/S  
Tooltype: PROCESS

Display the information structure associated with each DOS-process.

## CLI

Template: CLI/S  
Tooltype: CLI

Display the information structure associated with each CLI-task.

## CPU

Template: CPU/S  
Tooltype: CPU

Display several CPU usage values of each task.

---

## STACK

Template: STACK/S  
Tooltype: STACK

Display stack usage values.

## HELP

Template: HELP/S  
Tooltype: HELP

Display this text.

## VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.56 Ps Keywords

### PS KEYWORDS

#### General keywords

CPU%  
Recent (last minute) CPU usage percentage.

CSW  
Number of  
context switches  
for this task.

CTIME  
CPU time used by this task's terminated childtasks.

GID  
Group ID. Only available if MultiUser has been installed.

GROUP  
Group name. Only available if MultiUser has been installed.

ETIME  
Elapsed time since task was created.

IDLE  
How long task has been idle, without using any CPU time.

IVCSW  
Number of  
involuntary context switches

for this task.

**LSTART**

The exact task creation moment, including date.

**NAME**

Task name. Names of CLI processes are in square brackets.

**NICE**

Nice-value for this task.

**PID**

Process ID.

**PPID**

Parent Process ID.

**PRI**

Task's priority.

**RPRI**

Task's  
REAL priority  
.

**STACK**

Size of task's stack.

**STACK%**

Percentage of stack used.

**STACKUSE**

How many bytes of stack are currently used.

**START**

The time when task was created.

**STATE**

This will be one of these:

run task is running  
ready task is ready to run  
wait task is waiting

These states are very rare:

invalid something is really wrong with this task  
added task has just been added  
except task is in exception state  
removed task is being removed

**TIME**

Used CPU time.

**TIME%**

Percentage of CPU time used from total available CPU time.

**TTIME**

CPU time used by this task and its terminated childtasks.

---

## TYPE

Task type. This can be:

```
task    task
proc    process
cli     interactive CLI (always a process)
bcli   background CLI (always a process)
```

## UID

User ID. Only available if MultiUser has been installed.

## USER

User name. Only available if MultiUser has been installed.

## VOCSW

Number of  
voluntary context switches  
for this task.

## Task-structure keywords

These keywords display the entire Exec task-structure and aren't probably very interesting if you're not a programmer. They are listed in the same order as they appear in the structure, see <exec/tasks.h>.

## PRI

Task priority. (tc\_Node.ln\_Pri)

## TASK

Task-structure address.

## TNAME

Task name. Not same as NAME, which might be the CLI command name.  
(tc\_Node.ln\_Name)

## FLAGS

Task's flags: (tc\_Flags)

```
P process
T task
p proctime
e has etask-extension
S stackchk
x except
s switch-vector in use
l launch-vector in use
```

## STATE

Task state. See above. (tc\_State)

## NESTING

Interrupt disabled nesting / Task disabled nesting.  
(tc\_IDNestCnt, tc\_TDNestCnt)

## SIGALLOC

Signals allocated. (tc\_SigAlloc)

## SIGWAIT

Signals task is waiting for. (tc\_SigWait)

SIGRECV

Signals task has received. (tc\_SigRecvd)

SIGEXCEPT

Signals task will take excepts for. (tc\_SigExcept)

TRAPALLOC

Traps allocated. (tc\_TrapAlloc)

TRAPABLE

Traps enabled. (tc\_TrapAble)

EXCEPTDATA

Points to except data. (tc\_ExceptData)

EXCEPTCODE

Points to except code. (tc\_ExceptCode)

TRAPDATA

Points to trap data. (tc\_TrapData)

TRAPCODE

Points to trap code. (tc\_TrapCode)

STACKPTR

Stack pointer. (tc\_SPReg)

STACKLOW

Stack lower bound. (tc\_SPLower)

STACKUPR

Stack upper bound + 2. (tc\_SPUpper)

SWITCH

This vector is called when task is losing CPU. (tc\_Switch)

LAUNCH

This vector is called when task is getting CPU. (tc\_Launch)

USERDATA

For use by the task. (tc\_UserData)

From V36 upwards Exec has been able to allocate a structure called ETask, which stands for 'extended task'. The tc\_TrapAlloc and tc\_TrapAble are replaced with tc\_ETask pointer and these two values are moved to the ETask structure. This structure is not currently used, but you can display the structure with these keywords:

ETASK

ETask address. (tc\_ETask)

ET\_PARENT

Pointer to task-structure. (tc\_ETask->et\_Parent)

ET\_ID

---

ID unique to this task. (tc\_ETask->et\_UniqueID)

ET\_RESULT1

First result. (tc\_ETask->et\_Result1)

ET\_RESULT2

Result data pointer (AllocVec). (tc\_ETask->et\_Result2)

#### Process-structure keywords

These keywords display the entire DOS process-structure and aren't probably very interesting if you're not a programmer. They are listed in the same order as they appear in the structure, see <dos/dosextens.h>.

SEGLIST

Array of seg lists used by this process. (pr\_SegList)

PRSTACK

Process stack size. (pr\_StackSize)

GLOBVEC

Global vector for this process (BCPL). (pr\_GlobVec)

CLINUM

CLI task number or zero if not a CLI. (pr\_TaskNum)

PRSTACKUPR

Pointer to high memory end of process stack. (pr\_StackBase)

RESULT2

Value of secondary result from last call. (pr\_Result2)

CURDIR

Lock associated with current directory. (pr\_CurrentDir)

CIS

Current CLI Input Stream. (pr\_CIS)

COS

Current CLI Output Stream. (pr\_COS)

CONTASK

Console handler process for current window. (pr\_ConsoleTask)

FSTASK

File handler process for current drive. (pr\_FileSystemTask)

CLI

Pointer to CommandLineInterface. (pr\_CLI)

RETADDR

Pointer to previous stack frame. (pr\_ReturnAddr)

PKTWAIT

Function to be called when awaiting message. (pr\_PktWait)

---

## WINDOW

Window for error printing. (pr\_WindowPtr)

## HOMEDIR

Home directory of executing program. (pr\_HomeDir)

## PRFLAGS

Flags telling dos about process. (pr\_Flags)

- s freeseplist
- d freecurrdir
- c freecli
- i closeinput
- o closeoutput
- f freeargs

## EXITCODE

Code to call on exit of program or NULL. (pr\_ExitCode)

## EXITDATA

Passed as an argument to pr\_ExitCode. (pr\_ExitData)

## ARGS

Arguments passed to the process at start. (pr\_Arguments)

## SHELLPRIV

For the use of the current shell. (pr\_ShellPrivate)

## CES

Error stream - if NULL, use pr\_COS. (pr\_CES)

## CLI-structure keywords

These keywords display the entire CLI-structure and aren't probably very interesting if you're not a programmer. They are listed in the same order as the appear in the structure, see <dos/dosextens.h>.

## IOERR

Value of IoErr from last command. (cli\_Result2)

## SETNAME

Name of current directory. (cli\_SetName)

## COMMDIR

Head of the path locklist. (cli\_CommandDir)

## RETCODE

Return code from last command. (cli\_ReturnCode)

## COMMAND

Name of current command. (cli\_CommandName)

## FAIL

Fail level (set by Failat). (cli\_FailLevel)

## PROMPT

Current prompt (set by Prompt). (cli\_Prompt).

---

STDINPUT  
Default (terminal) CLI input. (cli\_StandardInput)

CURINPUT  
Current CLI input. (cli\_CurrentInput)

EXECUTE  
Name of EXECUTE command file. (cli\_CommandFile)

CLITYPE  
Interactive or background. (cli\_Interactive, cli\_Background)

CUROUTPUT  
Current CLI output. (cli\_CurrentOutput)

DEFSTACK  
Stack size to be obtained in long words. (cli\_DefaultStack)

STDOUTPUT  
Default (terminal) CLI output. (cli\_StandardOutput)

MODULE  
Seglist of currently loaded command. (cli\_Module)

#### Executive internal task-structure keywords

These keywords display some parts of the Executive internal task information structure, which is created for each task.

I use these for debugging and they aren't very useful for anybody else, so I don't explain these here.

BABY  
BABYNEXT  
BABYPREV  
BFLAGS  
CHILDREN  
CPU  
CPU2  
CSWBAL  
CURUTICKS  
DADDYUSAGE  
MONEY  
NEXTCHILD  
PARENT  
PREVCHILD  
RCS  
RUSAGE  
THASHNEXT  
THASHPREV  
UTICKS  
WATCH



## 1.57 Pstree

### PSTREE

~~~~~

Display all tasks in a tree-like structure.

OPTIONS DESCRIPTION

This client displays the task tree, the child-parent relationships. You can easily see from the output what tasks have been created by some other task. Pictures tell more than thousand words, so here:

```
+---trackdisk.device
+---gvpscsi.device
+---RAM
+---XH0
+---[QMouse]
|   +---[Pstree]
|   |   `---CON
|   `---[Make]
|       +---CON
|       `---[sh]
|           `---[ccl]
+---[CyberCron]
`---input.device
```

The root of the tree is at left. Tasks that branch from here don't have a known parent, i.e. they have been created before Executive server was started. To see a more complete tree, you must start the server in s:startup-sequence, just after SetPatch is run. Even then a few tasks are already running.

You can see that QMouse has created two tasks, Pstree and Make. Both are CLI processes, so they have created a CON process. The Make task has also created a shell, which has started a C-compiler.

If you are using AmigaGuide V39 or newer, you can see that the Pstree and Make tasks are displayed in italics. This means that the task which created them has terminated. In a case like this they are not left orphans, they are transferred to their 'grandparents'.

With the SHOWPIDS-option you can display task PIDs:

```
+---[QMouse] (5)
|   +---[Pstree] (9)
|   |   `---CON (224)
|   `---[Make] (8)
|       +---CON (236)
|       `---[sh] (1)
|           `---[cpp] (10)
...

```

Now you can root the tree to a specific PID. This is actually a process group, but in Executive they both mean the same:

```
Pstree PGRP=8
```

```
`---[Make] (8)
  +---CON (236)
    `---[sh] (1)
      `---[gcc] (10)
```

EXAMPLES

```
Pstree
```

1.58 Pstree options

PSTREE OPTIONS

PGRP

```
Template: R=PGRP/N
Tooltype: PGRP
```

Root the task tree to a specific process group.

SHOWPIDS

```
Template: SP=SHOWPIDS/S
Tooltype: SHOWPIDS
```

Display PIDs in the output.

IBM

```
Template: IBM/S
Tooltype: IBM
```

If you're using IBM style font, try this option, it uses graphical characters from this font to produce nicer output.

PLAIN

```
Template: PLAIN/S
Tooltype: PLAIN
```

Don't use ASCII characters in output.

BACKSLASH

```
Template: BS=BACKSLASH/S
Tooltype: BACKSLASH
```

The ` character may not look too good with all fonts, so with this option you can change it to `.

DELAY

Template: DELAY/N/K
 Tooltype: DELAY

Delay (in seconds) before outputting the tree.

HELP

Template: HELP/S
 Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
 Tooltype: -

Display version number.

1.59 Renice

RENICE

~~~~~

Change task's nice-value.

#### OPTIONS DESCRIPTION

Renice is used to increment the nice-value of a task, or multiple tasks at once.

The maximum nice-value is +20, making the task 'nice' to others, i.e. it will be given less CPU time than to a task with nice-value -20, which is the minimum value.

I want to stress that the value given to Renice is INCREMENT, not the final value. You can use negative increments to get lower values.

Renice can affect multiple tasks at once:

```
Renice 5 PID=10,20 NAME=DH\* USER=root GROUP=staff PGRP=273
```

The following tasks will be affected by this command:  
 (NOTE: Depending on the shell you use, the ` may not be needed in front of the asterisk.)

- tasks with PID 10 and 20
- tasks with name starting with 'DH' (case insensitive).
- all tasks owned by the root user. (MultiUser only)
- all tasks owned by the staff group. (MultiUser only)
- all tasks that belong to process group 273.

The new nice-value will be old + 5.

If you don't specify any task, the nice-value of the current task (usually a CLI process) will be affected.

There's no need to use the NICE- and PID-keywords, Renice works without them too:

```
Renice -10 278
```

This will add -10 to the nice-value of the task with PID 278.

The PGRP-option is sometimes very useful. Let's say you have a cron-utility installed in your system (I have). Cron is a program that runs in background and executes commands at specific times, as defined by user. I use a cron to take backups every day of important files and to check viruses. Pstree shows that I have four cronjobs running:

```
+---[CyberCron] (4)
| +---Virus_Checker(C) (235)
| +---[find] (1)
| +---[LhA] (10)
| `---[tar] (11)
...
```

These use a lot of CPU time and the scheduler keeps their priority lower, but we could still make them 'nicer'. That's simple:

```
Renice 20 PGRP=4
```

This will affect the CyberCron task and all its childtasks and also all tasks created by CyberCron from now on.

When MultiUser is installed, only the super-user may use negative nice-value increments. If you don't want to require root privileges for Renice, set the U-bit using MProtect.

## EXAMPLES

Add 10 to the current task's nice-value.

```
Renice 10
```

Force the nice-value to -20 for 'RexxMaster'. Because the maximum nice-value is +20, we must subtract at least 40 from it to always get to -20. If the nice-value would be 0, this would still be ok, because Renice makes sure that the nice-value doesn't go below -20.

```
Renice -40 name=RexxMaster
```

## 1.60 Renice options

### RENICE OPTIONS

~~~~~

NICE

Template: NICE/N
Tooltype: NICE

Increment of nice-value for the specified tasks.

PID

Template: P=PID
Tooltype: PID
Default: Current task

PIDs of the tasks whose nice-value is to be incremented. You can list multiple PIDs here, separate them with a comma, for example 'PID=10,248,332'.

NAME

Template: N=NAME/K
Tooltype: NAME

Names of the tasks whose nice-value is to be incremented. You can list multiple names here, separate them with a comma, for example 'NAME=DH0,DH1,DH2,*.device'. You can use * as a wildcard.

USER

Template: U=USER/K
Tooltype: USER

Names of the users whose every task's nice-value will be incremented. You can list multiple user names here, separate them with a comma, for example 'USER=frodo,pippin,gandalf'. You can also use UIDs. This option isn't available if MultiUser hasn't been installed.

GROUP

Template: G=GROUP/K
Tooltype: GROUP

Names of the groups whose every task's nice-value will be incremented. You can list multiple group names here, separate them with a comma, for example 'NAME=staff,students'. You can also use GIDs. This option isn't available if MultiUser hasn't been installed.

PGRP

Template: R=PGRP/K
 Tooltype: PGRP

PGRPs of the process groups whose nice-value will be incremented. All tasks in the process group will be affected. You can list multiple PGRPs here, separate them with a comma, for example 'PGRP=254,298,578'.

HELP

Template: HELP/S
 Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
 Tooltype: -

Display version number.

1.61 Sa

SA

~~~~~

Display system accounting statistics.

OPTIONS  
 DESCRIPTION

Sa displays a summary of accounting records and optionally writes this summary to disk. By default, per-command statistics will be displayed. The number of times a command has been executed, total CPU time used and total elapsed time. If you're using MultiUser, you can also get per-user statistics.

Sa maintains two summary files, 'acct.procsum' and 'acct.usersum'. These contain accounting records according to command name and user ID, respectively.

The per-command statistics output looks like this:

| COMMAND     | CALLS | CPUTIME | CPUTIME% | ELAPSED |
|-------------|-------|---------|----------|---------|
| NComm30     | 11    | 1h50m   | 23.39%   | 8h39m   |
| lzx         | 36    | 1h09m   | 14.73%   | 2h31m   |
| Mosaic      | 16    | 19m45s  | 4.18%    | 1h49m   |
| locate.sort | 6     | 10m57s  | 2.32%    | 23m33s  |
| csH         | 167   | 8m48s   | 1.87%    | 39h30m  |
| rCs         | 676   | 1m08s   | 0.24%    | 7m12s   |
| ed          | 703   | 41.43s  | 0.15%    | 18m40s  |
| newsyslog   | 6     | 3.27s   | 0.01%    | 29.48s  |
| RX          | 309   | 1.17s   | 0.00%    | 42.90s  |

```

Assign          32      0.96s      0.00%      1.59s
rlog            5      0.91s      0.00%     12.52s
ACUSeeMe_020    1      0.15s      0.00%      3.58s
stack          327      0.00s      0.00%      0.36s
...

```

For example, 'Mosaic' has been executed 16 times, 19 minutes 45 seconds of CPU time was used, which is 4.18% of all commands in the summary. Mosaic has been in system for 1 hour and 49 minutes.

The per-user statistics output looks like this (if MultiUser has been installed):

```

USER          COMMANDS  CPUTIME  CPUTIME  ELAPSED
-             8634      7h43m   97.85%   70h34m
root          13        9m14s   1.95%   22m06s
petrin        9         55.48s  0.20%   4m06s

```

Most of the commands, 8634, were executed without logging in to the MultiUser system, so the username is '-'. These commands have used 7 hours 43 minutes of CPU time. The 'root' user has used 1.95% of the CPU time and root's commands were in the system for 22 minutes 6 seconds.

It's possible to sort the summary in different ways. The default is to sort the summary by used CPU time, but it's also possible to sort by elapsed time, by command name, by average command execution time or by number of times a command was executed.

When you run Sa it will read the accounting file created by Acct, and the summary files. The new accounting records will be added to the summary. The MERGE option instructs Sa to write the summary back to disk and delete the current accounting file. The summary will be displayed on screen. To see per-user summary use the USER-option.

If you have multiple accounting summary files, you can merge them with Sa, for example:

```

Sa PFILE=acct.procsum.1,acct.procsum.2 UFILE=acct.usersum.1,
  acct.usersum.2 MERGE

```

This will read the 's:acct'-file, 's:acct.procsum', 's:acct.usersum' and the four files named above and merge them to 's:acct.procsum' and 's:acct.usersum'. The 's:acct'-file will be deleted. This picture shows what the command does:

```

s:acct.procsum -----
acct.procsum.1 ----- \
acct.procsum.2 ----- \ \                               .----> s:acct.procsum
                        \ \ \                               /
                        .-----./
s:acct ----->| Sa MERGE |-----> display summary on screen
                \-----/
                / / /
s:acct.usersum -----' / /                               \-----> s:acct.usersum
acct.usersum.1 -----' /
acct.usersum.2 -----'

```

If you want to run Sa automatically every day, you need a cron-program, for example CyberCron. You can get it from Aminet, in the util/time directory.

Sa doesn't require that the server is running.

Sa asks the Acct-daemon to write all unpurged records to disk so they will be included in the summary.

CTRL-C will stop the output.

#### EXAMPLES

Show a per-command summary:

Sa

This will join all records to a single group called \*\*\*other:

Sa JOINLONELY THRESHOLD=999999

## 1.62 Sa options

#### SA OPTIONS

##### ACCTFILE

Template: F=ACCTFILE  
Tooltype: ACCTFILE  
Default: s:acct

The accounting file name.

##### SUMMARYDIR

Template: SDIR=SUMMARYDIR/K  
Tooltype: SUMMARYDIR  
Default: s:

The directory where the summary files 'acct.procsun' and 'acct.usersun' are written.

##### AFILE

Template: AFILE/K  
Tooltype: AFILE

This is like the ACCTFILE-option, but you can list several accounting files here, separated by a comma. For example: 'AFILE=acct.1,acct.2'.

##### PFILE

---



Template: PFILE/K  
Tooltype: PFILE

This is used when merging multiple summary files. You can specify several procsum-summary files, separated by a comma. For example: `'PFILE=ap.1,ap.2,ap.3'`.

#### UFILE

Template: UFILE/K  
Tooltype: UFILE

This is used when merging multiple summary files. You can specify several usersum-summary files, separated by a comma. For example: `'UFILE=au.1,au.2,au.3'`.

#### MERGE

Template: MERGE/S  
Tooltype: MERGE

Merge the acct-files and all given summary files and write the new summary files 'acct.procsum' and 'acct.usersum' to SUMMARYDIR (default directory for summary files is s:).

#### NOACCT

Template: NOACCT/S  
Tooltype: NOACCT

Don't read the accounting-file (s:acct).

#### NOSUMMARY

Template: NOSUMMARY/S  
Tooltype: NOSUMMARY

Don't read the summary files. This will output a summary of the accounting-file (s:acct).

#### USER

Template: U=USER/S  
Tooltype: USER

Display the by user -accounting summary (acct.usersum).

#### JOINLONELY

Template: JL=JOINLONELY/S  
Tooltype: JOINLONELY

Join all commands that have been executed only one time to a group called '\*\*\*other'. See the THRESHOLD-option also.

This only affects the displayed summary.

---

## THRESHOLD

Template: TH=THRESHOLD/N/K  
Tooltype: THRESHOLD  
Default: 1

Used with JOINLONELY-option. If command has been executed only as many times as you specify here, it will be joined to a group called `\*\*\*other`. For example, `JOINLONELY THRESHOLD=10` will join all commands that have been executed 10 times or less to a group `\*\*\*other`.

## QUIET

Template: QUIET/S  
Tooltype: QUIET

Don't display the summary on screen.

## NOHEADER

Template: NH=NOHEADER/S  
Tooltype: NOHEADER

Don't display the header.

## SORTBYCALLTIME

Template: SCAT=SORTBYCALLTIME/S  
Tooltype: SORTBYCALLTIME

Sort the summary by average command execution time.

## SORTBYNAME

Template: SN=SORTBYNAME/S  
Tooltype: SORTBYNAME

Sort the summary by command name.

## SORTBYCALLS

Template: SC=SORTBYCALLS/S  
Tooltype: SORTBYCALLS

Sort the summary by number of times each command has been executed.

## SORTBYCTIME

Template: SCT=SORTBYCTIME/S  
Tooltype: SORTBYCTIME

Sort the summary by CPU time.

## SORTBYETIME

Template: SE=SORTBYETIME/S

---

Tooltype: SORTBYETIME

Sort the summary by elapsed time.

REVERSE

Template: RV=REVERSE/S

Tooltype: REVERSE

Reverse sort.

HELP

Template: HELP/S

Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S

Tooltype: -

Display version number.

## 1.63 Timer

TIMER

~~~~~

Time command execution.

OPTIONS

DESCRIPTION

The Timer client executes a command and measures its CPU usage. If Timer is started without a command name or it's started from Workbench, it will open a window where you can drop program icons.

Arguments can be given for the timed command as usual. If you're using a custom shell like csh or sksh, it's possible to run commands under that shell. Several clients use this feature, read about it from [here](#)

.

Timer provides normal or verbose output. The normal output consists of one line of information, it will be displayed after the command has been executed. For example:

```
time 0:00:04.79 + child 0:00:37.83 = total 0:00:42.62, elapsed 0:01:48.98
```

This means that the command consumed 4.79 seconds of CPU time. Its childtasks (i.e. the tasks that it created) used 37.83 seconds of CPU

time. This number includes the childtasks that were still running when the main command terminated. The command and its childtasks used 42.62 seconds of CPU time. 1 minute and 48.98 seconds of real time elapsed while the command was running.

The verbose output looks like this:

```
Task time                0:00:05.27 ( 3%)
Childtasks time         0:02:48.79 ( 97%)
Running childtask time  0:00:00.00 ( 0%)
=====
TOTAL USED CPU TIME    0:02:54.06 (100%)

Elapsed time           0:03:55.30
CPU-usage              73.97%
```

This command used 5.27 seconds of CPU time, which is 3% of the total CPU time used by the command and its childtasks. Those childtasks that terminated before the main command used 2 minutes and 48.79 seconds of CPU time, which is 97% of the total used CPU time. No childtasks were running when the main command terminated. 3 minutes 55.30 seconds of real time elapsed while the command was running. Of this time, this command and its childtasks used 73.97% of the total available CPU time.

Timing accuracy is 1/1000 seconds. The CPU time values are very accurate, they don't include the overhead that goes to task creation and loading it from disk. Neither does the elapsed time include any overhead, it's the time that elapses between AddTask() and RemTask() calls.

Timer can be used even if the server isn't running. Then only the elapsed time can be calculated.

When you start Timer from Workbench, or if you have Workbench running and start Timer from CLI without the command name, it will open a window on the Workbench screen. You can drop program icons to this window.

In normal mode, Timer will output the total CPU time used by the command and all its childtasks, elapsed time and CPU usage. You can switch to verbose mode by selecting the menu item Settings->Verbose. The produced output is identical to what is explained above. Window's titlebar displays the number of commands currently being executed.

If the command that you want to run doesn't have an icon, Timer will ask CLI arguments for it.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

EXAMPLES

This will start Timer and open a window on Workbench screen. If you don't have Workbench running, this won't work.

Timer

This times the c:Dir command:

```
Timer c:Dir
```

This times the c:Dir command and gives verbose output:

```
Timer VERBOSE c:Dir
```

As above, but appends the output to a file T:timer.out

```
Timer VERBOSE OUTPUT=T:timer.out APPEND c:Dir
```

1.64 Timer options

TIMER OPTIONS

OUTPUT

```
Template: O=OUTPUT/K
Tooltype: OUTPUT
```

This is used to redirect the output of Timer to a file.
For example:

```
Timer OUTPUT=T:timer.out dir
```

will output to T:timer.out file.

APPEND

```
Template: A=APPEND/S
Tooltype: APPEND
Default:  overwrite
```

Used with the OUTPUT-option. Instead of overwriting the specified file,
append to it.

VERBOSE

```
Template: V=VERBOSE/S
Tooltype: VERBOSE
Default:  output timing information on one line
```

Tell Timer to give verbose output.

STDOUT

```
Template: STDOUT/S
Tooltype: STDOUT
Default:  output timing information to stderr
```

By default, the timing information is outputted to standard error stream, which can't be redirected. By using this option, the timing information will be outputted to standard output stream.

LEFT

Template: LEFT/N/K
Tooltype: LEFT

Offset of the left edge of the window relative to screen.

TOP

Template: TOP/N/K
Tooltype: TOP

Offset of the top edge of the window relative to screen.

SHELL

Template: S=SHELL/K
Tooltype: SHELL
Default: Run command under standard Amiga shell

If you're using a custom shell, you can specify here a
template
that is
used when starting a command under that shell.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

COMMAND

Template: C=COMMAND/F
Tooltype: COMMAND

The command to be timed and its arguments.

1.65 Top

TOP

~~~~~

---

Display and update information about the tasks using most of the CPU time.

OPTIONS  
DESCRIPTION

Top displays the "Top 10" tasks using most CPU time. You can specify how many tasks are displayed.

Top will also display system load averages, number of tasks (sleeping, running), CPU usage, current time and available memory.

Top opens a window where it displays this information. The information will be updated periodically (by default every second).

It's also possible to run Top in a shell-window so it can be used from remote terminal. To only display the information once, use the ONCE-option. To run Top in a shell-window so that it updates its display use the STDOUT-option. You need a terminal that supports clear screen and cursor positioning commands. If you want to redirect Top's output to a file, use the ONCE-option instead of STDOUT. (To see why, try starting Top with STDOUT-option and redirect the output to a file. Let it run for a while and break Top with CTRL-C. Now see what the file contains. This might even be useful.)

When Top is running in a shell-window, it reads commands from the terminal and acts upon them accordingly. You need to press Enter after the command. These commands are available:

|        |                                                                                                                                                                                                                                                                               |
|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| h or ? | Print help text.                                                                                                                                                                                                                                                              |
| s or u | Change update interval (seconds).                                                                                                                                                                                                                                             |
| n      | Change number of tasks to display.                                                                                                                                                                                                                                            |
| i      | Toggle the displaying of idle tasks.                                                                                                                                                                                                                                          |
| d      | Redraw the display.                                                                                                                                                                                                                                                           |
| r      | Renice tasks. You'll be asked for nice-value increment and one or more PIDs. For example, entering "15 4,5,369" will increment nice-values of tasks 4,5 and 369 by 15. If MultiUser has been installed, then only the super-user may specify a negative nice-value increment. |
| k      | Kill tasks, i.e. send a CTRL-C break signal to them. You'll be asked for one or more PIDs of tasks to kill. With MultiUser, only the super-user may kill other users' tasks.                                                                                                  |
| q      | Quit.                                                                                                                                                                                                                                                                         |

This is an example of Top's display:

---

```
load averages: 0.63, 0.37, 0.10                      15:51:36
32 tasks: 30 sleeping, 2 running                      CPU usage: 18.1%
chip: 758K fast: 5451K virtual: 0K total: 6209K largest: 3966K
```

| PID | USERNAME | PRI | NICE | STATE | TIME | CPU    | NAME             |
|-----|----------|-----|------|-------|------|--------|------------------|
| 10  | petrin   | -89 | 0    | ready | 0:06 | 58.75% | [lha]            |
| 216 | -        | 4   | 0    | wait  | 0:13 | 18.42% | suprascsi.device |
| 201 | -        | 20  | 0    | wait  | 0:22 | 2.44%  | input.device     |
| 1   | root     | -70 | 0    | wait  | 0:07 | 1.74%  | [Quickgrab]      |
| 9   | -        | -51 | -15  | run   | 0:01 | 0.67%  | [Top]            |
| 6   | -        | 126 | 0    | wait  | 0:01 | 0.46%  | Executive        |
| 202 | -        | 10  | 0    | wait  | 0:00 | 0.32%  | DH2              |
| 211 | -        | 5   | 0    | wait  | 0:05 | 0.14%  | CON              |
| 3   | -        | 5   | 0    | wait  | 0:01 | 0.12%  | [ASwarm]         |
| 223 | -        | 10  | 0    | wait  | 0:01 | 0.11%  | DH0              |

Load averages are similar to what is displayed by  
Uptime

.

There are 32 tasks in the system, 30 of them are sleeping, 2 are running or ready. The current CPU usage is 18.1%, this is for the last second. The next line displays available memory. Available virtual memory is also displayed (GigaMem or VMM).

Task's CPU usage is a weighted average over the last minute.

PID is process identifier. USERNAME is displayed if MultiUser has been installed. PRI is task priority and NICE is task's nice-value. STATE is wait, ready or run. TIME is the amount of CPU time the task has used. NAME is the task name. If it's in square brackets, it's a CLI command name.

The menu item Project->New will redraw the display. The menu item Settings->Update sets the update interval. If you set the Settings->No idle tasks menu item, idle tasks won't be displayed. The Settings->Stay front keeps Top's window in front of all other windows.

If you double-click a task, Top will ask you if you want to send a break signal to that task. If MultiUser has been installed, then only super-user may break other users' tasks.

When you press the zoom-gadget, the window's height is reduced so that only the header is displayed.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

## EXAMPLES

Top

Like above, but show 15 tasks:



Top 15

Show as many tasks as possible:

Top MAX

Just output the information and then exit:

Top ONCE

Run Top in the current shell-window, press CTRL-C or 'q' + ENTER to quit:

Top STDOUT

Don't show idle tasks, update every 5 seconds:

Top NOIDLE UPDATE=5

Use a different font:

Top FONT=courier FONTSIZE=13

## 1.66 Top options

TOP OPTIONS

TASKS

Template: T=TASKS/N  
Tooltype: TASKS  
Default: 10

Number of tasks to display.

MAX

Template: MAX/S  
Tooltype: MAX

Show as many tasks as possible.

NOIDLE

Template: NI=NOIDLE/S  
Tooltype: NOIDLE  
Default: Show idle tasks also

Don't show idle tasks.

UPDATE

---

Template: UD=UPDATE/N/K  
Tooltype: UPDATE  
Default: 1

The update interval (in seconds).

#### FONT

Template: F=FONT/K  
Tooltype: FONT  
Default: System default font

Font to be used in Top's window. You might want to use a small font, so you can display more tasks. Top can only use mono-spaced, non-proportional fonts.

#### FONTSIZE

Template: FS=FONTSIZE/N/K  
Tooltype: FONTSIZE

If you specify a font, you should also specify the font size.

#### BGCOLOR

Template: BGC=BGCOLOR/N/K  
Tooltype: BGOLOR

Background color. (0-255)

#### TEXTCOLOR

Template: TXTC=TEXTCOLOR/N/K  
Tooltype: TEXTCOLOR

Text color. (0-255)

#### ZIP

Template: ZIP/S  
Tooltype: ZIP

Open Top's window so that only the header is displayed.

#### STDOUT

Template: S=STDOUT/S  
Tooltype: STDOUT

Don't open a window, display the information in the current shell-window.

#### BATCH

Template: BATCH/S  
Tooltype: BATCH

Disable reading keyboard when information is displayed in a terminal.

## ONCE

Template: ONCE/S  
Tooltype: ONCE

Don't open a window, just output the information and exit.

## UIDS

Template: UIDS/S  
Tooltype: UIDS

Display UIDs instead of user names.

## NOTITLEBAR

Template: NTB=NOTITLEBAR/S  
Tooltype: NOTITLEBAR

Don't display a titlebar for the window.

## STAYFRONT

Template: SF=STAYFRONT/S  
Tooltype: STAYFRONT

Keep the window in front of all other windows.

## PUBSCREEN

Template: PS=PUBSCREEN/K  
Tooltype: PUBSCREEN  
Default: default public screen

Name of the public screen where the window is to be opened.

## LEFT

Template: LEFT/N/K  
Tooltype: LEFT

Offset of the left edge of the window relative to screen.

## TOP

Template: TOP/N/K  
Tooltype: TOP

Offset of the Top edge of the window relative to screen.

## HELP

Template: HELP/S  
Tooltype: HELP

---

Display this text.

VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.67 Stat

STAT

~~~~~

Display miscellaneous statistics.

OPTIONS
DESCRIPTION

This is a sample output from the Stat-client:

Executive 1.20

Registered to: Petri Nordlund [#1]

Boottime: Tuesday 11-Apr-95 14:44:59
Uptime: 27 mins
Load average: 0.41 0.69 0.48
Total used CPU time: 0:15:49.55 (62.88%)
Total idle CPU time: 0:09:20.45 (37.12%)
Context switches: 243841

Tasks created/finished: 7/3
Tasks running/sleeping: 2/27
Number of clients/handlers: 1/0
Number of known windows: 2

Scheduler:	Super	Timer:	CIA-B
Catch range:	-105/2	Dynamic range:	-100/-50
Time resolution:	1/1000s	Watched tasks:	6

Seconds before purge:	100	Max records in memory:	8
Unpurged records:	0	Accounting daemon pid:	7
Records created:	247		

Stat outputs miscellaneous information about the system and Executive:

Registered to

If your name isn't here, then shame on you! ;-)

Boottime

The time when your system was booted.

Uptime

How long your system has been up.

Load average
System load averages for 1, 5 and 15 minutes.

Total used CPU time
How much CPU time has been used and how many percentages that is from uptime.

Total idle CPU time
How long time the CPU has been idle.

Context switches
Number of context switches been made.

Tasks created/terminated
Number of tasks added and removed from the system.

Tasks running/sleeping
Number of tasks currently running or sleeping.

Number of clients/handlers
Number of clients currently connected to the server. A handler is created by a client that needs to update its display frequently (e.g. ALoad and Top).

Number of known windows
How many opened windows have a known owner.

Scheduler
Currently used
scheduler
.

Timer
Currently used
timer
.

Catch range & Dynamic range
See the
server documentation
Time resolution
The accuracy of CPU usage measurements.

Watched tasks
How many task's are currently being watched
.

Accounting is deactivated
Tells you if accounting is off.

Seconds before purge
How many seconds to wait (maximum) before writing records to disk.

Max records in memory
Maximum number of records held in memory at once.

Unpurged records
Records currently in memory

Accounting daemon pid
PID of the Acct daemon.

Records created
Total number of accounting records created.

EXAMPLES

Stat

1.68 Stat options

STAT OPTIONS

~~~~~

#### HELP

Template: HELP/S  
Tooltype: HELP

Display this text.

#### VERSION

Template: VERSION/S  
Tooltype: -

Display version number.

## 1.69 Uptime

### UPTIME

~~~~~

Display current time, length of time the system has been up, number of users, and system load averages.

OPTIONS DESCRIPTION

Uptime displays current time, length of time the system has been up, number of users (with MultiUser), and load averages over the last 1, 5, and 15 minutes on a single line. If started from Workbench, a window will be opened and the information will be displayed in window's titlebar. This information will be updated by default every second.

Unlike other Uptime utilities for Amiga, this one displays correct Uptime, there's no need to run any programs or daemons when the system is booted.

Uptime is displayed as 'up 10 mins', 'up 1:43' (one hour, 43 minutes) or 'up 7 days'.

The number of users currently logged in is got from multiuser.library. It's only shown if MultiUser has been installed.

The system load averages are displayed as three numbers, for example:

```
load: 2.12, 0.67, 0.15
```

This means that over the last minute, approximately 2.12 tasks have been running or ready to run. The higher the number, the higher the load in your computer is, i.e. the more tasks there are fighting for CPU time. The other two numbers are for 5 and 15 minutes, respectively.

When Uptime is run from Workbench, it opens a window and displays this information in the window's titlebar. You can choose the update interval from menus, as well as what information should be displayed.

The Settings->Stay front menu item, when set, keeps the Uptime window in front of all other windows.

The Project->Save config menu item will save current settings to the program's icon. This option is available only when the program is started from Workbench.

EXAMPLES

```
Uptime
```

```
Uptime NOTIME NOUSERS
```

```
Uptime WINDOW
```

```
Uptime WINDOW NOTIME NOUSERS NOUPTIME LEFT=50
```

1.70 Uptime options

UPTIME OPTIONS

```
~~~~~
```

WINDOW

```
Template: W=WINDOW/S
```

```
Tooltype: NOWINDOW
```

```
Default: Use window if started from Workbench
```

If Uptime is started from Workbench, it will open a window to show the information. This can be overridden with the NOWINDOW-tooltype. If you start Uptime from CLI and want it to use a window, use the WINDOW-option.

NOTIME

```
Template: NOTIME/S
```

```
Tooltype: NOTIME
```

Don't display the current time.

NOUPTIME

Template: NOUPTIME/S
Tooltype: NOUPTIME

Don't display Uptime.

NOUSERS

Template: NOUSERS/S
Tooltype: NOUSERS

Don't display the number of users currently logged in.

NOLOAD

Template: NOLOAD/S
Tooltype: NOLOAD

Don't display system load averages.

UPDATE

Template: UD=UPDATE/N/K
Tooltype: UPDATE
Default: 1

Update interval in seconds, used only when the information is displayed in a window.

STAYFRONT

Template: SF=STAYFRONT/S
Tooltype: STAYFRONT

Keep the window in front of all other windows.

LEFT

Template: LEFT/N/K
Tooltype: LEFT

Offset of the left edge of the window relative to screen.

TOP

Template: TOP/N/K
Tooltype: TOP

Offset of the top edge of the window relative to screen.

PUBSCREEN

Template: PS=PUBSCREEN/K
Tooltype: PUBSCREEN
Default: default public screen

Name of the public screen where the window is to be opened.

HELP

Template: HELP/S
Tooltype: HELP

Display this text.

VERSION

Template: VERSION/S
Tooltype: -

Display version number.

1.71 About custom shells

ABOUT CUSTOM SHELLS

~~~~~

These two clients have to execute CLI commands:

- \* Nice
- \* Timer

Commands are normally run under the standard Amiga CLI, but if you're using a custom shell, you can run commands under it, and even run that shell's internal commands. This is accomplished with an environmental variable called EXECUTIVESHELL.

This variable contains a shell command which is used to start the command under a custom shell. Here are some examples:

```
csh:    csh -c %c
sksh:   sksh -c %c
ksh:    ksh -c %c
```

So, if you're using csh-shell, set the EXECUTIVESHELL to "csh -c %c" with these commands:

```
setenv EXECUTIVESHELL "csh -c %c"
copy ENV:EXECUTIVESHELL ENVARC:
```

The %c is substituted with the command name and its arguments.

You can now run internal shell commands and use shell's aliases and even pipes, if they're supported by the shell. You may have to use \| instead of | if you want to use pipes this way because the command line is parsed twice.

If you need to use the % character in the command, use %% instead.

I'm using csh and I have this command in my s:.cshrc:

```
cd A:Com/D1
```

This will change the directory to A:Com/D1 so that when I start a new csh, it will always be in the same directory. Now, I must use this EXECUTIVESHELL variable:

```
csh -c "cd %d ; %c"
```

The %d is substituted with the current directory. This is needed because otherwise csh would try to start all commands from A:Com/D1-directory, not from the directory I have run Timer or Nice client. For example, if I'm in directory WORK:Programs and do this:

```
WORK:Programs> Timer lha x TMP:archive.lha
```

Timer will execute this command:

```
csh -c "cd WORK:Programs ; lha x TMP:archive.lha"
```

This ensures that the archive is extracted to the directory WORK:Programs.

If you're using some other shell that is mentioned above, tell me what EXECUTIVESHELL variable you are using so I can add it here.

---