SetDST

Stefan Falke

COLLABORATORS

	TITLE : SetDST	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Stefan Falke	August 8, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

SetDST

1.1 SetDST User Guide

SetDST 1.8 (10.3.2000)

==========

User Guide

==========

1.2 LEGALALESE

LEGALESE

If you want to use this software read the COPYRIGHT, the DISCLAIMER and any other documentation that can be found here before taking any further steps.

1.3 COPYRIGHT

COPYRIGHT

1.4 DISCLAIMER

DISCLAIMER

1.5 Introduction

Introduction

In most countries of the world local time changes two times a year: from
local standard time, also known as Wintertime, to local daylight saving time

(DST), also known as Summertime, and back.
SetDST helps you to keep up with the DST switches and will perform any
necessary action automagically.
It may be run in the background if you wish, so if you never switch your
computer off it will make sure that you always have the right system
time--hopefully, though.
In addition to that SetDST creates and manages up to 4 environment variables,
TZONE, TZ, TIMEZONE and YAM_TZ, which enables other applications to make use
of the time zone names and GMT offsets stored within.
It is able to patch the default locale by user request.

1.6 Requirements

Requirements
° AmigaOS 2.04 or higher

1.7 Installation

Installation
There is no special installation process required to run SetDST --
unpack the archive to a place you like and run the program.
However, if you want a more convenient method of installing the program
there is an Installer script provided with this release, called
Install-SetDST.
If you double-click its icon only those files will be installed you
really need and you may choose other options like installing SetDST so
that it will be run automatically during startup.
If you run SetDST for the first time your system time should be set to your
current local time.
In addition to that your locale settings, maintained by the Locale Preferences
editor, should be set to your Country and Time Zone so that SetDST is able
to select the correct DST scheme for your area.
Both settings are important if you run SetDST with the builtin default
settings -- the experienced user may override the default settings.
If the environment variable TZ is already present, its DST part should
reflect the time in your zone, either standard time or DST.

1.8 Usage

Usage

SetDST can either be run from the CLI or from the Workbench.

Running from the CLI

Running from the Workbench

Aborting the program

The DST.dat format

The Timezone data format

Environment variables

1.9 Running from the CLI

Running from the CLI

Format: SetDST [TZDATA <tzdata>] [ZONE <zone>]
[SETTZ] [TZFIX MODE1|MODE2] [SASFIX] [SETTIMEZONE]
[SETYAMTZ] [ADDNAME] [SETLOCALE] [REMOVEENV]
[BACKGROUND] [NOSAVE] [NOUSE] [BATTCLOCKONLY]
[NOASK] [NOREQ] [QUIET] [NOLOG]
[REFRESH <seconds>] [NOWATCHENV] [CX_PRIORITY <priority]
Template: TZDATA/K,ZONE/K,
SETTZ/S,TZFIX/K,SASFIX/S,SETTIMEZONE/S,
SETYAMTZ/S,ADDNAME/S,SETLOCALE/S,REMOVEENV/S,
BACKGROUND/S,NOSAVE/S,NOUSE/S,BATTCLOCKONLY/S,
NOASK/S,NOREQ/S,QUIET/S,NOLOG/S,
REFRESH/N/K,NOWATCHENV/S,CX_PRIORITY/N/K

1.10 Running from the CLI

TZDATA

When no TZDATA argument is given, the program will read and parse the file DST.dat which contains time zone specific information for various countries and other geographical regions.

By default, the program uses the current locale settings (Country and Time Zone) to retrieve the correct timezone record from DST.dat.

SetDST will search for DST.dat in the s drawer of the current directory or in S:.

With TZDATA you can specify a different path for the program to look for DST.dat. The value has to be a complete path including the filename, 'DH0:data/DST.dat' for example.

In addition to that TZDATA may contain a complete timezone record in the Timezone data format which will be used instead of records from DST.dat.

1.11 Running from the CLI

ZONE

By specifying ZONE you can force SetDST to search for and use a specific record in DST.dat .

ZONE may contain a record identifier or a standard timezone name and a GMT offset to identify a record.

If SetDST is unable to retrieve a timezone record from DST.dat or a wrong timezone string is given with TZDATA the US rules are used to switch to and from DST, which is the first Sunday in April at 02:00 and the last Sunday in October at 02:00 with DST beeing one hour ahead of standard time.

In this case the names of the time zones will default to 'GMT' and the GMT offset is set to the current Locale Time Zone offset.

1.12 Running from the CLI

SETTZ

By default SetDST stores the current timezone names and GMT offsets in the environment variable TZONE only.

With SETTZ you tell the program to create the environment variable TZ as well, which has the format as documented by the SAS/C compiler environment.

1.13 Running from the CLI

TZFIX

Unfortunately there are at least three different versions of how the environment variable TZ should look like.

If an application produces wrong results when evaluating a TZ created by SetDST using SETTZ, you can use TZFIX to create the TZ variable in two additional formats.

With TZFIX you have to specify one of currently two values, MODE1 or MODE2 .

TZFIX is only used in combination with SETTZ.

1.14 Running from the CLI

SASFIX

is only used in combination with SETTZ.

If set it will have the same effect as TZFIX=MODE1.

Please use TZFIX=MODE1 as this argument may disappear in future versions.

1.15 Running from the CLI

SETTIMEZONE
With SETTIMEZONE you tell SetDST to create the environment variable TIMEZONE
which will hold a RFC 822 compliant timezone string.

1.16 Running from the CLI

SETYAMTZ
With SETYAMTZ you tell the program to create the environment variable
YAM_TZ , which has the same format as TIMEZONE.
If this variable is present when you use the EMail client YAM to write mail,
its contents will be added to the 'Date' line of the mail body.

1.17 Running from the CLI

ADDNAME
is only used in combination with SETTIMEZONE and SETYAMTZ.
If specified the name of the timezone will be added to the timezone
strings of the TIMEZONE or YAM_TZ environment variables.

1.18 Running from the CLI

SETLOCALE
As the current AmigaOS doesn't offer a documented standard way of setting
DST, the best way for an application to evaluate the current DST mode for
creating time specific information would be to look out for a special
environment variable like TZONE or TZ.
However, many applications do not behave this way and simply use the
current Time Zone value set with the Locale Preferences program which
will obviously produce wrong results when DST is in effect.
To support this kind of programs you can tell SetDST to set the GMT
offset of the default locale to the value calculated by SetDST with
the SETLOCALE argument.
Please note that SETLOCALE will only work if BACKGROUND is specified
too. If SetDST is stopped from running the original GMT offset of the
default locale will be restored.
When specifying SETLOCALE you should not set NOWATCHENV. This will allow
SetDST to react to any changes beeing made to the locale preferences--

SetDST might behave incorrectly otherwise!
NOTE: SETLOCALE will change the GMT offset system wide for any
application wich uses the default locale. Having other programs running
which do DST calculations by themselves will very likely cause them to
behave incorrectly so this argument should be used with care!

1.19 Running from the CLI

REMOVEENV
will cause SetDST to behave different:
It will only delete environment variables maintained by SetDST.
This option may be useful if you decide not to use SetDST anymore or you
do not want to remove environment variables by hand for a different
reason.
By default SetDST will only remove TZONE.
If you specify SETTZ, SETTIMEZONE and/or SETYAMTZ together with
REMOVEENV, the corresponding variables will be deleted as well.
The variables will be deleted from ENVARC: and ENV: unless you specify
NOSAVE to remove them from ENV: only.

1.20 Running from the CLI

BACKGROUND
If you run SetDST without the BACKGROUND argument it will check once what time
it is, do any necessary system time changes, write all required environment
variables and then exit.
If your system is powered on 24 hours/7 days a week you can let SetDST
watch for changes in the background by specifying BACKGROUND.
This way SetDST will run as a Commodity and the behaviour of it can be managed
with the 'Commodities Exchange' program.
Please refer to the Notes section for possible side effects when specifying
BACKGROUND.

1.21 Running from the CLI

NOSAVE
With NOSAVE you prevent SetDST from writing any changes to the battery-backed
clock and environment variables will only be written to ENV: and not to
ENVARC:
In combination with REMOVEENV environment variables will only be deleted
from ENV:

1.22 Running from the CLI

NOUSE

When NOUSE is specified SetDST will NOT change system time if necessary and
will NOT write any environment variables at all.
It will only tell you what it wants to do via CLI or logfile messages.

1.23 Running from the CLI

BATTCLOCKONLY

Under some conditions it can be dangerous to set back system time (see Notes).
With BATTCLOCKONLY you can force SetDST to only change the battery-backed
clock and leave the system time untouched.

1.24 Running from the CLI

NOASK

By default SetDST will bring up a requester that has to be satisfied
before it changes system time.
Whith NOASK these changes will be performed without prior confirmation.

1.25 Running from the CLI

NOREQ

With NOREQ you tell SetDST not to bring up any requester, it overrides NOASK.

1.26 Running from the CLI

QUIET

By default SetDST writes informative messages to a console window.
If you do not want to see these messages, you should specify QUIET.

1.27 Running from the CLI

NOLOG

Those informative messages will also be written to t:SetDST.log unless you
tell SetDST not to do so by specifying NOLOG.

1.28 Running from the CLI

REFRESH
is especialy useful if you have to set back system time while SetDST
is running which is always a thing that might cause problems for a numerous
kind of other programs as well (see Notes).
If system time is set back beyond the previous date for a DST switch SetDST
wouldn't be able to catch the next switch based on the new time.
With REFRESH you can specify the duration of the interval in seconds SetDST
looks for any changes done to system time from outside SetDST.
So with the default value of 60, SetDST looks every 60 seconds for changes and
if there were any that would affect the current DST settings, it will perform
all necessary action.
60 seconds should be a good compromise between accuracy and system
load--choosing shorter intervals might slow down other running programs.
However, if you want SetDST to use almost no processor time at all while in
the background you can disable this feature with a REFRESH value of 0.
NOTE: you can always force SetDST to reevaluate the current settings by either
quitting and restarting it or by deactivating and reactivating it through
Commodities Exchange.

1.29 Running from the CLI

NOWATCHENV
SetDST mainly uses TZONE to store the current DST setting.
If you change this environment variable from outside SetDST while SetDST is
running, the program will notice that by default and perform any necessary
changes immediately.
SetDST will also monitor changes beeing made to the Locale settings and will
act in the same way.
With NOWATCHENV this feature can be disabled.

1.30 Running from the CLI

CX_PRIORITY
lets you set the priority of SetDST running in the Commodities
Exchange environment. It accepts values from -128 to 127, the higher the
number, the higher the priority of the SetDST commodity.

1.31 Running from the Workbench

Running from the Workbench

All of the CLI arguments also exist as Workbench ToolTypes with the same meaning.

If you want to run SetDST during startup, the recommended way is to copy SetDST and SetDST.info to the SYS:WBStartup drawer.

If you do so, you should add the tooltype DONOTWAIT so that the startup sequence does not wait until SetDST has finished.

You should also copy DST.dat to the S: drawer.

The Installer script offers an option to copy all files to the right place for running SetDST during startup if you select 'Average' or 'Expert' mode.

1.32 Aborting the program

Aborting the program

If you run SetDST without the BACKGROUND argument there is no need to abort the program--it will run once making all necessary changes and then exit.

If you run SetDST with the BACKGROUND argument you have several ways aborting the running program:

° Open the Commodities Exchange window and select the 'Remove' gadget.

° SetDST can only run once on your system. If you try to start it again it will refuse to open and will also force the already running SetDST to exit.

° Send a break signal (Ctrl-C) to the SetDST task and the program will exit.

° If you run SetDST from the CLI, press the 'Ctrl-C' keys in the CLI window.

1.33 DST.dat format

DST.dat format

DST.dat contains timezone records for various countries and other geographical regions.

Each line represents a timezone record which mainly consists of a string in the Timezone data format .

In addition to that it may be preceded by a record identifier which will allow to classify and/or closer identify a record.

Record identifiers and timezone strings are separated by a ';' character.

They are used, for example, when the current locale settings are to be used to retrieve a record or if you specify the ZONE argument.

Comments in DST.dat are identified by a '#' character, blank lines and white spaces are ignored.
The records in DST.dat are parsed from top to bottom--the first record that matches will be used.
To reduce parsing time or avoid problems with duplicate record identifiers you can copy your timezone record to the beginning of the file.
Example:
USA;EST5EDT,M4.1.0/2,M10.5.0/2 # Eastern Time Zone
In this record 'USA' is the record identifier, 'EST5EDT,M4.1.0/2,M10.5.0/2' is the timezone string and ' Eastern Time Zone' is a comment.
It will be selected by SetDST if your Locale settings are set to 'United_states' and '5 Hours from GMT' and you do not specify a different timezone string via TZDATA and no ZONE argument is given.
This record will also be chosen if you specify 'USA' with the ZONE argument (because the record for the Eastern Time Zone is the first record in DST.dat which matches 'USA') or if you specify 'USA5' or 'EST5' (if you want to make sure that only the record with a matching identifier/zone name and GMT offset will be selected).
Alternatively, to use this time zone data you can supply 'EST5EDT,M4.1.0/2,M10.5.0/2' with the TZDATA argument.
If you would specify 'EST' with the ZONE argument the timezone record for Eastern Australia would be selected because it is the first record which matches 'EST'.
This is how the complete DST.dat DST.dat currently looks like.

1.34 Timezone data format

Timezone data format
The timezone string specifies the names of timezones and the GMT offsets of these zones SetDST uses for creating environment variables as well as the rules of these zones that will tell SetDST when to switch to and from DST.
The string is based upon the POSIX.1 TZ format and has the following form:
std offset dst[offset],date[/time],date[/time]
The spaces are for clarity only and should be ommited!
where:
std and dst
are 3 or more characters specifying the local standard and daylight saving time (DST) zone names.
offset

is of the form [-]hh:[mm[:ss]] and specifies the offset west of GMT.
If there is no offset following dst daylight saving time is one hour ahead
of standard time.
date[/time],date[/time]
specifies the beginning and end of DST.
If this is absent, the US rules for DST are used.
date
is of the form Mm.n.d for the dth day of week n of month m of the
year (1<=m<=12, 1<=n<=5, 0 (Sunday)<=d<=6(Saturday)).
Week 1 is the first week in which day d appears and '5' stands for the
last week in which day d appears.
time
takes the form hh:[mm[:ss]] and defaults to 02:00 for the beginning and
end of DST.
Example:
CET-1CEST-2,M3.5.0/2,M10.5.0/3
means standard time is called CET and is one hour east of GMT, DST is
called CEST and, if it is in effect, is 2 hours east of GMT.
DST starts on the last Sunday in March at 02:00 and ends on the last Sunday
in October at 03:00.
PST8PDT,M4.1.0/2,M10.5.0/2
means standard time is called PST and is 8 hours west of GMT, DST is
called PDT and, if it is in effect, is 7 hours west of GMT.
DST starts on the first Sunday in April at 02:00 and ends on the last Sunday
in October at 02:00.

1.35 Environment variables

Environment variables
SetDST uses environment variables to store and retrieve the current DST
settings and to allow other applications to make use of it.
They will be stored by default in ENV: and in ENVARC: unless you specify the
NOUSE and/or NOSAVE arguments.
TZONE
The main variable managed by SetDST is TZONE.
Its format is very similar to the TZDATA format:
std offset[dst [offset]]
The spaces are for clarity only and should be ommited!
where:

std and dst
are 3 or more characters specifying the local standard and daylight saving time (DST) zone names.
offset
is of the form [-]hh:[mm[:ss]] and specifies the offset west of GMT.
If there is no offset following dst daylight saving time is one hour ahead of standard time.
If the dst offset part is absent, SetDST assumes that system time represents standard time, if it is present, system time means daylight saving time.
Offset always specifies the absolute offset of the zone from GMT so if standard time is in use, you should use the std offset and if DST is in effect, the dst offset should be used.
Note that this is different to the timezone environment variable TZ!
Examples:
EST5
standard time is in effect, your timezone is called EST and you are 5 hours west of GMT.
CST-9:30CST-10:30
DST is in effect, your timezone is called CST and you are 10 hours, 30 minutes east of GMT.
CET-1CEST
DST is in effect, your timezone is called CEST and you are 2 hours east of GMT.
TZ
If you specify the SETTZ argument, SetDST will write the TZ environment variable as well.
By default it will be in the format used by the SAS/C compiler environment:
aaabbb[ccc]
where:
aaa
is a three-letter abbreviation for the local standard time zone.
bbb
is of the form -23 to 24 and specifies the offset west of GMT for local standard time in hours.
ccc
is a three-letter abbreviation for the local DST zone.
It is present if DST is in effect and local time is one hour ahead of standard time.

If TZFIX=MODE1 is specified along with SETTZ, TZ will be in the following
format:
aaabbb
aaa
is a three-letter abbreviation for the local standard or daylight
saving time zone.
bbb
is of the form -23 to 24 and specifies the offset west of GMT for local
standard time or daylight saving time in hours.
Please note that when specifying MODE1 it is not possible for applications to
determine by TZ whether standard or daylight saving time is in effect!
If TZFIX=MODE2 is specified along with SETTZ, TZ will be in the following
format:
aaabbb[ccc]
aaa
is a three-letter abbreviation for the local standard time zone.
bbb
is of the form -23 to 24 and specifies the offset west of GMT for local
standard time or daylight saving time in hours.
ccc
is a three-letter abbreviation for the local DST zone.
It is present if DST is in effect.
Examples:
CET-1
means standard time is in effect, your timezone is called CET and you
are 1 hour east of GMT.
EST5EDT
means DST is in effect, your timezone is called EDT, and you are 4 hours
west of GMT.
EDT4 (created in combination with TZFIX=MODE1)
means your timezone is called EDT and you are 4 hours west of GMT.
It is unknown whether DST is in effect or not.
EST4EDT (created in combination with TZFIX=MODE2)
means DST is in effect, your timezone is called EDT and you are 4 hours
west of GMT.
TIMEZONE, YAM_TZ
If you specify the SETTIMEZONE argument, SetDST will write the TIMEZONE
environment variable as well.
If you specify the SETYAMTZ argument, SetDST will write the YAM_TZ environment

variable as well.
Both will contain a RFC 822 compliant time string in the following format:
[+/-]hhmm
If ADDNAME is specified along with these variables, TIMEZONE and YAM_TZ
will be in the following format:
[+/-]hhmm (aaa)
where aaa is a three or more letter abbreviation for the local
standard or daylight saving time zone.
Examples:
A TZONE containing CST-9:30CST-10:30 will produce the variable
TIMEZONE or YAM_TZ containing '+1030', a TZONE containing EST5 will
produce '-0500'.
In combination with ADDNAME a TZONE containing PST8PDT7 will
produce '-0700 (PDT)'.

1.36 Tools

IsDST
IsDST is a small CLI utility which allows to check whether DST is
currently in effect on the system.
It will parse TZONE and, as a fallback, TZ and will return WARN (5) when
DST rules and OK (0) when standard time is in effect or if neither TZONE
nor TZ are present.
It will return FAIL (20) in case of a serious internal allocation error.
A possible way to use IsDST would be from an AmigaDOS script:
IsDST
IF WARN
ECHO "DST is in effect"
ELSE
ECHO "Standard time is in effect"
ENDIF
CheckDST
CheckDST is a small shell script which shows an easy way to check whether DST
is currently in effect on the system.

1.37 Notes

Notes

SetDST's main job is to change the Amiga system time and the
battery-backed clock when it thinks the time for a DST switch has come.
Under some conditions changing the time backwards or forwards while other
programs are running might cause some serious problems:
All programs that compute time measurement or depend on a continuous
increasing system time may come into dire straits -- they can produce
wrong results, may behave in an unpredictable way or even crash the
computer, especially when the time is set backwards.
If you have such kind of programs running on your system and would still
like to use SetDST there are several possibilities to avoid problems:
° You can force SetDST to only set the battery-backed clock and not the
system time by specifying the BATTCLOCKONLY argument.
° If you do not want SetDST to have the ability to change system time
while the system is running, do NOT specify the BACKGROUND argument.
Instead you should run SetDST once during startup as early as possible
before you run any program that might do timing operations.
° You should NOT specify the NOASK argument and you should NOT specify
the NOREQ argument.
This way every time SetDST wants to change the time, it asks you for
confirmation before - if you have programs running at that moment that
perform timing routines, select 'No' to prevent SetDST from changing
the time.

1.38 Examples

Examples

1. SetDST

SetDST will retrieve the timezone record from DST.dat based on the current
locale settings.
It will check once if local time is DST or standard time, see how it matches
with the environment variables, adjust system time and variables if necessary
and then exit.
If it wants to change system time and environment variables, it asks the user
for confirmation.
It will create the environment variable TZONE.

2. SetDST ZONE="EST5"

SetDST will retrieve the timezone record from DST.dat matching the time zone
name "EST" and a GMT offset of ’5’ hours west of GMT.
It will check once if local time is DST or standard time, see how it matches
with the environment variables, adjust system time and variables if necessary
and desired by the user and then exit.
It will create the environment variable TZONE.
3. SetDST TZDATA="CET-1CEST-2,M3.5.0/2,M10.5.0/3" BACKGROUND NOASK SETTZ
SETTIMEZONE SETYAMTZ REFRESH=0
SetDST will use the timezone string "CET-1CEST-2,M3.5.0/2,M10.5.0/3".
It will run in the background and watch out for the next DST switch.
If it wants to change system time and environment variables, it does not
ask the user for confirmation.
It will create the environment variables TZONE, TZ, TIMEZONE and YAM_TZ and
will never check and react automatically if system time has been set back.
4. SetDST NOUSE
SetDST will run once and will neither change system time nor any environment
variables.
5. SetDST REMOVEENV SETTZ
SetDST will run once and will delete the TZONE and TZ environment variables
from both ENVARC: and ENV:

1.39 Author

Author
SetDST is written by Stefan Falke.
If you have any comments about the program, should find any bugs or would
like to see special features to be added in the next release please feel
free to send me an email to the following address:
Stefan Falke <sfalke@gmx.de>

1.40 Feedback

Rating SetDST
Please rate SetDST.
To do so, send an EMail to
aminet-server@wuarchive.wustl.edu
with
RATE util/time/SetDST.lha <num>
in the body of the mail, where <num> is your rating from 0..10 with 10

being the best.

You can rate several programs in one mail.

DST.dat

If you have to add or change timezone records in DST.dat to get the correct values for your country or region please let me know of these changes.

I will accomodate DST.dat in future releases.

Language support

If you would like to have language support for a language which isn't supplied with this release, please let me know if you would like to make a translation yourself and I will contact you ASAP.

Please let me know *before* you start translating--there may be another person already doing the same job.

1.41 Acknowledgements

Acknowledgements

The language catalogs were translated by:

Dansk: Thomas Siefert <ziefert@gmx.net>

Español: Dámaso D. Estévez <amidde@arrakis.es>

Français: Didier Giron <girondid@fnac.net>

Italiano: Davide Zipeto <dawez@tiscalinet.it>

Norsk: Kimme Utsi <kimme@arcticnet.no> (ATO translation)

Deutsch: Me :-)

CheckDST was contributed by Nils Goers <nils@goers.line.org>

Many thanks to everyone else who contributed.

1.42 History

History

1.8 (10.3.2000)

° The logfile now displays years with four digits.

° OS35 icons have been added.

° Spanish language catalog added to distribution archive.

1.7 (10.09.99)

° Italian language catalog added to distribution archive.

° Various internal modifications (well, a few at least :).

1.6 (30.07.99)

° The argument TZFIX has been added to allow further variations of the TZ

environment variable to be created.

It can replace the SASFIX argument.

° CheckDST script added to distribution archive.

° In V1.5, DST.dat wasn't found if it was located in 'PROGDIR:s' - this has been fixed.

° Danish language catalog added to distribution archive.

1.5 (18.5.99)

° The argument ADDNAME has been added which will add the name of the current time zone to the timezone string of the environment variables TIMEZONE and YAM_TZ.

° SetDST will not react to changes anymore beeing made to TZONE or the Locale settings when it is inactive.

° Norwegian language catalog added to distribution archive.

° Installer script has been modified.

° IsDST binary added to distribution archive.

1.4

° Internal release.

1.3 (26.4.99)

° It is now possible to force SetDST to only change the battery-backed clock by specifying the new argument/tooltype BATTCLOCKONLY.

° It is now possible to set the default locale to the current offset from GMT determined by SetDST by specifying the new argument/tooltype SETLOCALE.

° It is now possible to remove all or single environment variables maintained by SetDST by specifying the new argument/tooltype REMOVEENV.

° The environment variable TZ can now be written in a different form to support applications which work around the broken SAS/C time functions by specifying the new argument/tooltype SASFIX.

° Installer script added to distribution archive.

° French language catalog added to distribution archive.

° Added more messages when changing system time/battery-backed clock and when setting/deleting environment variables.

° SetDST now requires language catalogs V2.x.

1.1, 1.2

° Internal releases.

1.0 (24.3.99)

° First public release.