rexxgls

Anders Wegge Jakobsen

COLLABORATORS

	TITLE : rexxgls	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Anders Wegge Jakobsen	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

rexxgls

1.1 rexxlocaldates.library Documentation

```
                This is the hypertext documentation of rexxlocaldates. ↩
                   library

              (C) 1996, Anders Wegge Jakobsen


            Introduction
             What is this anyway?

               Author
             Who wrote it?

           Acknowledgments
             No, it's all features.

               History
             How many mistakes have he made so far?

               Future
             Does he intend to fix them?

                Bugs
             No, it's all features.


           Documentation
             How to use it.
```

1.2 Introduction

locale.library provides an ARexx host, which can be used for localizing ARexx scripts. This host does not provide access to all functions of locale.library, however. This library provides the missing functions,

but is not intended as an replacement for the ARexx host of
locale.library.

This library is freeware, ie. you may distribute it as much as you like, provided you don't charge anything for it. Distributors of PD CD-ROMs should contact me first, if they intend to include this on a CD-ROM. The Aminet collection and Fred Fish are excluded from this restriction.

Even though this is freeware, I'd like to have some sort of response (e-mail, postcard, a 6-pack of your local beer), if you use this library regulary.

1.3 Who made this? I did!

If you haven't already noticed it on the title page, I can inform you that this library was written by:

Anders Wegge Jakobsen, Age 26, no known vices.

If tou want to contact me, you should try one of the following methods, in order of appearance:

E-mail:

wegge@daimi.aau.dk - CS, Aarhus University.
wegge@hoa.ping.dk - Offline gateway, run by a Danish Amiga-BBS network.
awjakob@aabc.dk - Informatics, Aarhus Buisiness College. Dont use this, if you can avoid it. Mail tends to get lost, or the mailserver can't find itself or something like that.

Fidonet/BBS:

2:238/28@fidonet, or online at Sirius Cybernetics at +45-8675-5253

Snail-mail:

Anders Wegge Jakobsen
Bentesvej 49, st. tv.
8220 Brabrand
Denmark

1.4 Without whom I never would have succeded

I would like to thank the following persons for their help with this project:

Rolf Rotvel - Initial idea, testing.

Peter Kærså - Testing.

Preben Nielsen - ARexx information.

1.5 Major dates in the development

8/6 - 96 2.0 First public release.

4/6 - 96 1.12 Various bugs fixed, sent into test once more.

FIX: 2 ASCII 0's were added to the result from FormatDate().

FIX: OpenLocale() demanded an argument, even if the default locale were requested.

29/5 - 96 1.10 Completed, sent into beta testing.

17/5 - 96 1.0 First working release, with one function sent into early alpha testing.

May - 96 0.x Several internal versions, all failing in one way or another.

1.6 Coming features ...

Further development will depend on the amount of feedback i receive. Especially because I'm unable to come up with anything related to localization, which isn't already covered by either locale.library or rexxlocaldates.library.

1.7 Bugs? No, it's all features!

This release does not contain any known bugs. Otherwise I wouldn't release it. However, some bugs may remain, which I'd be glad to hear about, and fix. If you belive you have found a genuine bug, the please report it to me, with a complete description of how to reproduce it. Bugs I cannot reproduce will be very difficult to fix.

1.8 documentation

This Library provides six functions:

OpenLocale()

CloseLocale()

```
                GetLocaleString()

                GetLocaleVars()

                FormatDate()

                ParseDate()
                Before theese are available, the library should be added to ARexx ↩
                    '
library list:

 IF (~SHOW('L','rexxlocaldates.library')) THEN
   CALL ADDLIB('rexxlocaldates.library',0,-30,0)

 /* Use localized Arexx here */
```

1.9 Locale = OpenLocale(Name)

```
                 USAGE

    Locale = OpenLocale( Name )


  INPUTS

    Name -- The name of an locale. This Locale must be created by the
    Preferences editor (Prefs/Locale


  DESCRIPTION

    Before a locale can be used, it must be opened. This function will
    return a value, that must be used for any calls to the other
    functions of this library.

    If the specified locale cannot be opened, the users preferred
    locale will be instead. In this case, the value RC will be set
    to 5.

    Calls to this function must be paired with calls to CloseLocale().
    If a locale isn't closed when the script finished, the library
    will be unable to flush itslf from memory, and the associated
    memory will be lost.


  EXAMPLE

    l_Default = OpenLocale('')


  SEE ALSO
```

```
CloseLocale()
```

1.10 CloseLocale(Locale)

```
                USAGE

  CALL CloseLocale( Locale )


INPUTS

  Locale -- An already opened locale.


DESCRIPTION

  Releases an opened locale. Any locales still opened, when a script
  terminates wastes memory, and inhibits the library from flushing
  itself.


EXAMPLE

  CALL CloseLocale(l_Default)


SEE ALSO


              OpenLocale()
```

1.11 String = GetLocaleString(Locale, Number)

```
                USAGE

  String = GetLocaleString( Locale, Number )


INPUTS

  Locale  -- A locale returned from
             OpenLocale()
                 Number  -- An integer indicating which string is requested.


RESULT

  String  -- The requested string.
```

```
DESCRIPTION

  This function queries locale.library for one of it's localized
  strings. Currently (as of locale.library 40.4), there are 50
  strings defined. They are:

    1  -  7   Full names of weekdays, Sunday - Saturday.
    8  - 14   Abbreviated names of weekdays, Sunday - Saturday.
   15  - 26   Full names of months, January - December.
   27  - 38   Abbreviated names of months, January - December.
         39   Affirmative response (Yes).
         40   Negative response (No).
   41  - 42   12 hour time qualifiers (AM, PM)
   43  - 44   Hyphenation chars, Hard and Soft.
   45  - 46   Opening and closing quotes.
         47   Yesterday
         48   Today
         49   Tomorrow
         50   Future

  If the specified number is out of bounds, the returned string will
  be empty, an RC will be set at 5.


EXAMPLE

  FOR Days = 1 TO 7 DO
    SAY 'Day ' || Days || ' is ' || GetLocaleString( Locale, Days)
  END
```

1.12 CALL GetLocaleVars(Locale, STEM)

```
                USAGE

  CALL GetLocaleVars( Locale, STEM)


INPUTS

  Locale  -- An already opened locale.
  STEM    -- Basename of the stem to be filled.


DESCRIPTION

  The specified stem will be filled with the following values:

    STEM.LOCALENAME       -- Name of the locale.
    STEM.LANGUAGE         -- Preferred language.

    STEM.DATETIME         -- Format string for date and time.
    STEM.DATEFORMAT       -- Format string for date alone.
```

```
    STEM.TIMEFORMAT       -- Format string for time alone.

    STEM.SHORTDATETIME    -- Compact date and time.
    STEM.SHORTDATEFORMAT  -- Compact date alone.
    STEM.SHORTTIMEFORMAT  -- Compact time alone.

  The dateformats can be used as arguments for calls to FormatDate()
  and ParseDate.

SEE ALSO

              FormatDate()
              ,
              ParseDate()
```

1.13 String = FormatDate(Locale, Datestamp, Format)

```
                USAGE

  String = FormatDate( Locale, Datestamp, Format)

INPUTS

  Locale     -- An already opened locale.

  Datestamp  -- Basename of a stem containing the following fields:
                STEM.DAYS    - Days since 1/1-1978
                STEM.SECONDS - Seconds since midnight

                Current date and time can be obtained with:
                Now.DAYS    = DATE('I')
                Now.SECONDS = TIME('S')

  Format     -- A string specifying how the date should be formatted. All
                formatting codes start with %, followed by a single char.
                Any text in the string, which don't match a format code
                will be copied to the output. The codes are:

                      %a - abbreviated weekday name
                      %A - weekday name
                      %b - abbreviated month name
                      %B - month name
                      %c - same as "%a %b %d %H:%M:%S %Y"
                      %C - same as "%a %b %e %T %Z %Y"
                      %d - day number with leading 0s
                      %D - same as "%m/%d/%y"
                      %e - day number with leading spaces
                      %h - abbreviated month name
                      %H - hour using 24-hour style with leading 0s
                      %I - hour using 12-hour style with leading 0s
                      %j - julian date
                      %m - month number with leading 0s
                      %M - the number of minutes with leading 0s
```

```
                        %n - insert a linefeed
                        %p - AM or PM strings
                        %q - hour using 24-hour style
                        %Q - hour using 12-hour style
                        %r - same as "%I:%M:%S %p"
                        %R - same as "%H:%M"
                        %S - number of seconds with leadings 0s
                        %t - insert a tab character
                        %T - same as "%H:%M:%S"
                        %U - week number, taking Sunday as first day of week
                        %w - weekday number
                        %W - week number, taking Monday as first day of week
                        %x - same as "%m/%d/%y"
                        %X - same as "%H:%M:%S"
                        %y - Year using two digits with leading 0s
                        %Y - Year using four digits with leading 0s


RESULT

  String  -- A formatted date.


NOTES

  If no value is specified as Datestamp, the resulting string will be
  current date and time. If Format is omitted, the default format for the
  locale will be used.

SEE ALSO


               ParseDate()
```

1.14 Success = ParseDate(Locale, Datestamp, Format, String)

```
                USAGE

  Success = ParseDate( Locale, Datestamp, Format, String )


INPUTS

  Locale     -- An already opened locale.

  Datestamp  -- Basename of the stem to be filled.

  Format     -- A string describing how the date are expected to be
                formatted. The string is constructed similar to the string
                used for
              FormatDate()
             , but only the following %switches
                can be used:
                  %a %A %b %B %d %e %h %H %I %m %M %p %S %y %Y
```

```
  String     -- The date that should be parsed.

RESULTS

  If the string can be parsed according to the specified format, Datestamp
  will be filled in, in the same way as described for
              FormatDate()
             , and 1
  will be returned. Otherwise 0 will be returned, in which case Datestamp
  can contain any or no value at all.


SEE ALSO


               FormatDate()
```