

# gpxutils Group

John D. Baker

<https://github.com/bakerjd99/jacks/blob/master/geotagged/gpxutils.ijs>

SHA-256: 453a9c5b2e2ce2d3a68f70e3851c851e910fab07d81b006a042750def038970

June 18, 2022

## Contents

<b>gpxutils Overview</b>	<b>2</b>
gpxutils Interface . . . . .	2
<b>gpxutils Source Code</b>	<b>3</b>
<b>=: Index</b>	<b>25</b>

## gpxutils Overview

gpxutils is a [J script](#) that formats Garmin style waypoint [GPX files](#) from CSV files, Google Maps KML files, and the SmugMug [SQLite](#) `mirror.db` database. The resulting GPX files can be loaded into the Motion-GPS iPhone app and other GPS devices that import GPX data.

To run gpxutils inspect interface word comments.

gpxutils is generated from [JOD dictionaries](#) `gps` and `utils`.

*NB. open JOD dictionaries and generate gpxutils script*

```
load 'general/jod'  
od  ;:'gps utils'  
mls 'gpxutils'
```

A generated gpxutils script and sample `mirror.db` database are here:

- <https://github.com/bakerjd99/jacks/blob/master/geotagged/gpxutils.ijs>
- <https://github.com/bakerjd99/smugpyter/blob/master/testdata/mirror.zip/mirror.db>

## gpxutils Interface

<code>allrecent</code>	<code>[7]</code>	<i>all recent images from last waypoint generation</i>
<code>csvfrwpt</code>	<code>[10]</code>	<i>poi CSV text from waypoint text file</i>
<code>gpxfrmakml</code>	<code>[14]</code>	<i>gpx from Google maps kml</i>
<code>gpxfrmirror</code>	<code>[16]</code>	<i>extracts geotagged images from mirror_db and generates gpx</i>
<code>gpxfrpoicsv</code>	<code>[17]</code>	<i>converts poi csv files to gpx</i>
<code>gpxfrrecent</code>	<code>[19]</code>	<i>gpx from recent waypoints</i>

## gpxutils Source Code

```
NB.*gpxutils s-- generate gpx waypoint files from various
NB. sources.
NB.
NB. This group formats Garmin style waypoint gpx files from CSV
NB. files, my SmugMug sqlite mirror database, and Google map KML.
NB. The resulting gpx files can be loaded into the Motion-GPS
NB. iPhone app and other GPS devices that import gpx data.
NB.
NB. verbatim: interface words
NB.
NB. allrecent - all recent images from last waypoint generation
NB. csvfrwpt - poi CSV text from waypoint text file
NB. gpxfrmapkml - gpx from Google maps kml
NB. gpxfrmirror - extracts geotagged images from mirror_db and generates gpx
NB. gpxfrpoicsv - converts poi csv files to gpx
NB. gpxfrrecent - gpx from recent waypoints
NB.
NB. created: 2019dec11
NB. changes: -----
NB. 19dec18 added (allrecent)
NB. 22jun18 merged (gpxfrmapkml) and dependents

require 'data/sqlite regex'
coclass 'gpxutils'
```

*NB.\*dependents*

*NB. (\*)=: GPXFRKMLHEADER GPXHEADER GPXSMUGPLACEMARK GPXTRAILER*

*NB.\*enddependents*

GPXFRKMLHEADER=: (0 : 0)

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<gpx version="1.1"
  creator="J GPX from Google Maps KML script"
  xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  xmlns="https://www.topografix.com/GPX/1/1"
  xsi:schemaLocation="https://www.topografix.com/GPX/1/1/gpx.xsd">
<metadata>
<name>{{headername}}</name>
<desc>{{headerdescription}}</desc>
<link href="https://analyzethedatanotthedrivel.org/">
<text>Analyze the Data not the Drivel</text>
</link>
</metadata>
)
```

GPXHEADER=: (0 : 0)

```
<gpx xmlns="https://www.topografix.com/GPX/1/1"
  xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  creator="J Waypoints"
  version="1.1"
  xsi:schemaLocation="https://www.topografix.com/GPX/1/1/gpx.xsd">
<metadata>
```

```
<link href="https://www.jsoftware.com">
<text>J (gpxutils) last waypoint = {{date}}</text>
</link>
```

```
</metadata>
)
```

```
GPXSMUGPLACEMARK=: (0 : 0)
<wpt lat="{{latitude}}" lon="{{longitude}}">
<ele>0</ele>
<name>{{phototitle}}</name>
</wpt>
)
```

```
GPXTRAILER=: (0 : 0)
<extensions>
</extensions>
</gpx>
)
```

*NB.\*end-header*

*NB. get all images from mirror - select columns*

```
AllMirror_sql=: 'select Latitude, Longitude, RealDate, UploadDate, OnlineImageFile from OnlineImage'
```

*NB. carriage return character*

```
CR=: 13{a.
```

*NB. valid gpx name characters*

```
GPXNAMECHARS=: ' -()0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

*NB. get geotagged images from mirror - rows in desc upload date*

```
GpxGeotaggedMirror_sql=: 'select Latitude, Longitude, RealDate, UploadDate, OnlineImageFile from OnlineImage  
>..>e where Keywords like "%geotagged%"'
```

*NB. regular expression matching placeholder variables in html lists*

```
HTMLVARBPATTERN=: '{{[a-z]*}}'
```

*NB. interface words (IFACEWORDSgpxutils) group*

```
IFACEWORDSgpxutils=: <;._1 ' allrecent csvfrwpt gpxfrmapkml gpxfrmirror gpxfrpoicsv gpxfrrecent'
```

*NB. line feed character*

```
LF=: 10{a.
```

*NB. gpx file written by (gpxutils)*

```
MIRRORGPXFILE=: 'c:/pd/coords/gpx/geotagged smugmug images.gpx'
```

*NB. root words (ROOTWORDSgpxutils) group*

```
ROOTWORDSgpxutils=: <;._1 ' IFACEWORDSgpxutils ROOTWORDSgpxutils VMDgpxutils allrecent csvfrwpt gpxfrmapkml  
>..> gpxfrmirror gpxfrpoicsv gpxfrrecent write'
```

*NB. version, make count, and date*

```
VMDgpxutils=: '0.9.0';7;'18 Jun 2022 18:39:13'
```

*NB. retains string (y) after last occurrence of (x)*

```
afterlaststr=: ] }.~ #@[ + 1&(i:~)@([ E. ])
```

*NB. retains string after first occurrence of (x)*

```
afterstr=: ] }.~ #@[ + 1&(i.~)@([ E. ])
```

```
allrecent=: 3 : 0
```

*NB.\*allrecent v-- all recent images from last waypoint generation.*

*NB.*

*NB. monad: bt =. allrecent clMirrorDb*

*NB.*

*NB. trg=. 'c:/smugmirror/documents/xrefdb/mirror.db'*

*NB. allrecent trg*

*NB.*

*NB. dyad: bt =. clGpxFile gpxfrrecent clMirrorDb*

*NB.*

*NB. lastgpx=. 'c:/pd/coords/gpx/geotagged test images.gpx'*

*NB. lastgpx allrecent trg*

```
MIRRORGPXFILE allrecent y
```

```
:
```

```
waydate=. waystamp gpx=. read x NB. extract last waypoint date
```

*NB. the last upload date is shifted forward to partly compensate*

*NB. for the mixture of UTC and local dates. The times in the database*

*NB. come from many time zones and many timestamps are just approximations.*

```
sql=. AllMirror_sql , ' where UploadDate > date("",waydate,"", ''+16 hours'') order by UploadDate desc '
sql fst y
)
```

*NB. trims all leading and trailing blanks*

```
alltrim=: ] #~ [: -. [: (*. /\ . +. */ \) ' ' &=
```

*NB. signal with optional message*

```
assert=: 0 0" _ $ 13!:8^:((0: e. ])^ (12" _))
```

*NB. retains string before first occurrence of (x)*

```
beforestr=: ] {.~ 1&(i.~)@([ E. ])
```

```
betweenstrs=: 4 : 0
```

*NB.\*betweenstrs v-- select sublists between nonnested delimiters*

*NB. discarding delimiters.*

*NB.*

*NB. dyad: blcl =. (clStart;clEnd) betweenstrs cl*

*NB. bnl =. (nlStart;nlEnd) betweenstrs nl*

*NB.*

*NB. ('start';'end') betweenstrs 'start yada yada end boo hoo start ahh end'*

*NB.*

*NB. NB. also applies to numeric delimiters*

*NB. (1 1;2 2) betweenstrs 1 1 66 666 2 2 7 87 1 1 0 2 2*

```
's e'=. x
```



```
llst=. ((-#s) (|.!0) s E. y) +. e E. y
mask=. ~:/\ llst
(mask#llst) <;.1 mask#y
)
```

```
NB. boxes open nouns
boxopen=: <^(L. = 0:)
```

```
changestr=: 4 : 0
```

```
NB.*changestr v-- replaces substrings - see long documentation.
NB.
NB. dyad: clReps changestr cl
NB.
NB. NB. first character delimits replacements
NB. '/change/becomes/me/ehh' changestr 'blah blah ...'
```

```
pairs=. 2 {."(1) _2 [\ <;._1 x      NB. change table
cnt=._1 [ lim=. # pairs
while. lim > cnt=:cnt do.           NB. process each change pair
  't c'=. cnt { pairs               NB. /target/change
  if. +./b=. t E. y do.             NB. next if no target
    r=. I. b                         NB. target starts
    'l q'=. #&> cnt { pairs          NB. lengths
    p=. r + 0,+/\(<:# r)$ d=. q - 1 NB. change starts
    s=. * d                          NB. reduce < and > to =
    if. s = _1 do.
```

```

    b=. 1 #~ # b
    b=. ((1 * # r)$ 1 0 #~ q,l-q) (,r +/ i. l)} b
    y=. b # y
    if. q = 0 do. continue. end. NB. next for deletions
elseif. s = 1 do.
    y=. y #~ >: d r} b NB. first target char replicated
end.
y=. (c $~ q *# r) (,p +/i. q)} y NB. insert replacements
end.
end. y NB. altered string
)

```

```
csvfrwpt=: 3 : 0
```

```

NB.*csvfrwpt v-- poi CSV text from waypoint text file.
NB.
NB. monad: cl =. csvfrwpt clFile
NB.
NB. f=. 'C:\Users\baker\iCloudDrive\oznztrip\gps_oz_nz_2022.txt'
NB. p=. '.'&beforestr f
NB. t=. csvfrwpt f
NB. (toHOST t) write p, '.csv'
NB. g=. gpæfrpoicsv p, '.csv'
NB. (toHOST g) write p, '.gpæ'

NB. lines from text
ct=. < ;._2 tlf (read y) -. CR

```

*NB. waypoint names*

```
wn=. ':'&beforestr&.> ct
```

*NB. extract longitude and latitude*

```
lb=. |."1 <;._1"1 ' , ' ,&> -.&' '&.> (':'&afterstr)@('; '&beforestr)&.> ct
```

*NB. format comma delimited*

```
em=. 1 0 1 0 1  
lb=. alltrim&.> lb ,. wn  
tlf ctl ;"1 (<','') (<a;;I. -.em)} em (#^:_1)"1 lb  
)
```

*NB. character table to newline delimited list*

```
ctl=: }.@(:,@ (1&(",1)@(-.@(*./\"1@(&' '0]))) # ,@((10{a.)&(",1)@])
```

*NB. enclose all character lists in blcl in " quotes*

```
dblquote=: '''&,@:(,&'')&.>
```

```
eletags=: 4 : 0
```

*NB.\*eletags v-- encloses xml text (y) in xml element tag.*

*NB.*

*NB. dyad: clTag eletags clXml*

```
tag=. alltrim x  
'<',tag,'>',y,'</',tag,'>'<br>)</pre>
```

```
fmtmirrorgpx=: 3 : 0
```

```
NB.*fmtmirrorgpx v-- formats mirror_db sql query results as gpx.
```

```
NB.
```

```
NB. monad: fmtmirrorgpx btSqlDict
```

```
NB. insure any singletons are shaped
```

```
ix=. I. (0 {"1 y) e. ;:'RealDate UploadDate OnlineImageFile'
```

```
y=. (boxopen&.> (<ix;1){y) (<ix;1)} y
```

```
y=. (,&.> 1 {"1 y) (<a;1)} y
```

```
NB. quit if no data
```

```
if. +./ 0 = #&> 1 {"1 y do. '' return. end.
```

```
NB. !(*)=. Latitude Longitude RealDate UploadDate OnlineImageFile
```

```
(0 {"1 y)=. 1 {"1 y
```

```
NB. clean file names
```

```
names=. '['&beforestr@justfile&.> OnlineImageFile
```

```
names=. alltrim&.> names -.&.> names -.&.> <GPXNAMECHARS
```

```
'names cannot be null' assert -. 0 e. #&> names
```

```
NB. format latitude and longitude
```

```
wpt=. (<LF,'<wpt lat=') ,. (dblquote 8!:0 Latitude) ,. (<' lon=') ,. (dblquote 8!:0 Longitude) ,. <'>'
```

```
NB. format dates for gpx
```

```
RealDate=. alltrim@((,&'Z')@('+'&beforestr))&.> RealDate
```

```
UploadDate=. alltrim@((,&'Z')@('+'&beforestr))&.> UploadDate

NB. use real date unless empty else use upload date
ix=. I. 0 = #&> RealDate
RealDate=. (ix{UploadDate) ix} RealDate
wpt=. wpt ,. 'time'&eletags&.> RealDate

NB. waypoint names & descriptions
wpt=. wpt ,. _1 |."1 names ,"0 1 |. tags 'name'

NB. symbols
wpt=. wpt ,. <'sym' eletags 'waypoint'
wpt=. wpt ,. <'</wpt>'

NB. last waypoint upload date
gpxhead=. ('/{date}]/', }: ;0{UploadDate) changestr GPXHEADER

NB. return gpx
gpxhead,(;wpt),LF,'</gpx>'
)

fsd=: 4 : 0

NB.*fsd v-- fetch sqlite dictionary array.
NB.
NB. dyad: clSql fsd clDb
NB.
NB. trg=. 'c:/smugmirror/documents/xrefdb/mirror.db'
```

```
NB. sql=. 'select ImageKey, OriginalWidth, OriginalHeight, OnlineImageFile, Keywords from OnlineImage'
NB. sql fsd trg
```

```
NB. require 'data/sqlite' !(*)=. sqlclose__db sqldict__db sqlopen_psqlite_
d [ sqlclose__db '' [ d=. sqldict__db x [ db=. sqlopen_psqlite_ y
)
```

```
fst=: 4 : 0
```

```
NB.*fst v-- fetch sqlite reads table.
```

```
NB.
```

```
NB. dyad: bt =. clSql fst clDb
```

```
NB.
```

```
NB. trg=. 'c:/smugmirror/documents/xrefdb/mirror.db'
```

```
NB. sql=. 'select ImageKey, OriginalWidth, OriginalHeight, OnlineImageFile, Keywords from OnlineImage'
```

```
NB. sql fst trg
```

```
NB. require 'data/sqlite' !(*)=. sqlclose__db sqlreads__db sqlopen_psqlite_
d [ sqlclose__db '' [ d=. sqlreads__db x [ db=. sqlopen_psqlite_ y
)
```

```
NB. get pure element text
```

```
geteletext=: ] betweenstrs~ [: tags [: alltrim [
```

```
gpxfrmapkml=: 3 : 0
```

```
NB.*gpxfrmapkml v-- gpx from Google maps kml.
```

```
NB.
NB. monad:  clGpx =. gpæfrmapkml clKml
NB.
NB.  NB. download Google map waypoints as kml
NB.  kml=. read 'c:/temp/arizona annular eclipse.kml'
NB.
NB.  NB. convert to gpx and save
NB.  gpx=. gpæfrmapkml kml
NB.  gpx write 'c:/temp/arizona annular eclipse.gpx'

NB. parse kml form waypoint table
dname=. ;'name' geteletext '<Placemark>' beforestr y
wpt=.   ;'Placemark' geteletext y
wpt=.  ('name' geteletext wpt) ,. <:_1&> ','&,&.> 'coordinates' geteletext wpt
hdr=.   ;:'phototitle longitude latitude'

NB. format gpx header
gpxstamp=. 'Waypoints: ',( "#wpt), ' GPX generated: ',timestamp''
gpxheader=. ('/{headername}}/',dname,'/{headerdescription}}/',gpxstamp) changestr GPXFRKMLHEADER
gpxtrailer=. GPXTRAILER

'idx pkml'=. HTMLVARBPATTERN patpartstr GPXSMUGPLACEMARK
rvarbs=. idx htmlvarbs pkml

msg=. 'all row variables must exist in data header'
msg assert */ rvarbs e. hdr
rows=. (#wpt) # ,: pkml
```

```
rows=. ((hdr i. <'phototitle'){1 wpt) (<a;;(rvarbs i. <'phototitle'){idx}) rows
rows=. ((hdr i. <'latitude'){1 wpt) (<a;;(rvarbs i. <'latitude'){idx}) rows
rows=. ((hdr i. <'longitude'){1 wpt) (<a;;(rvarbs i. <'longitude'){idx}) rows

gpxheader,(;rows),gpxtrailer
)

gpxfrmirror=: 3 : 0

NB.*gpxfrmirror v-- extracts geotagged images from mirror_db and generates gpx.
NB.
NB. monad: clGpx =. gpxfrmirror clMirrorDb
NB.
NB.   trg=. 'c:/smugmirror/documents/xrefdb/mirror.db'
NB.   gpx=. gpxfrmirror trg
NB.   (toHOST gpx) write 'c:/pd/coords/gpx/geotagged smugmug images.gpx'
NB.
NB. dyad: clGpx =. iaN gpxfrmirror clMirrorDb
NB.
NB.   10 gpxfrmirror trg

0 gpxfrmirror y NB. all waypoints default
:
NB. limit waypoints
sql=. GpxGeotaggedMirror_sql , ' order by UploadDate desc ' , ;(0<x){'';' limit ',":x
fmtmirrorgpx sql fsd y
)
```



```
gpxfrpoicsv=: 3 : 0
```

```
NB.*gpxfrpoicsv v-- converts poi csv files to gpx.
```

```
NB.
```

```
NB. This verb converts comma delimited point of interest (POI)
```

```
NB. *.csv files to Garmin compatible gpx files. Example POI files
```

```
NB. can be downloaded from:
```

```
NB.
```

```
NB. http://www.poi-factory.com/poifiles
```

```
NB.
```

```
NB. monad: clGpx =. gpxfrpoicsv clCsvfile
```

```
NB.
```

```
NB. gpx=. gpxfrpoicsv 'c:\pd\coords\poicsv\ca_park_m.csv'
```

```
NB.
```

```
NB. dyad: clGpx =. iaRows gpxfrpoicsv clCsvfile
```

```
NB.
```

```
NB. gpx=. 10 gpxfrpoicsv 'c:\pd\coords\poicsv\ca_park_m.csv'
```

```
0 gpxfrpoicsv y NB. format all waypoints default
```

```
:
```

```
NB. read csv file
```

```
csv=. parsecsv read y
```

```
if. 0<x do. csv=. (x<.#csv) {. csv end.
```

```
NB. sanity test latitude and longitude
```

```
lbcheck=. -. _9999 e. , _9999 "&> 0 1 {"1 csv
```

```
'invalid longitude latitude number representations' assert lbcheck
```

*NB. clean names*

```
names=. 2 {"1 csv  
names=. alltrim&.> names -.&.> names -.&.> <GPXNAMECHARS  
'names cannot be null' assert -. 0 e. #&> names
```

*NB. format latitude and longitude*

```
csv=. (dblquote 0 1 {"1 csv) (1 0)}"1 csv  
wpt=. (<LF,'<wpt lat=') ,. (0{"1 csv) ,. (<' lon=') ,. (1{"1 csv) ,. <'>'
```

*NB. times set to now*

```
wpt=. wpt ,. <'time' eletags nstamp=. gpxtimestamp 6!:0''
```

*NB. waypoint names & descriptions*

```
wpt=. wpt ,. _1 |."1 names ,"0 1 |. tags 'name'  
NB. wpt=. wpt ,. _1 |."1 (alltrim&.> 3 {"1 csv) ,"0 1 |. tags 'desc'
```

*NB. symbols*

```
wpt=. wpt ,. <'sym' eletags 'waypoint'  
wpt=. wpt ,. <'</wpt>'
```

*NB. waypoint format date*

```
gpxhead=. ('/{date}}/', ):nstamp) changestr GPXHEADER
```

*NB. return gpx*

```
gpxhead,(;wpt),LF,'</gpx>'
```

```
)

gpxfrrecent=: 3 : 0

NB. *gpxfrrecent v-- gpx from recent waypoints.
NB.
NB. monad: clGpx =. gpxfrrecent clMirrorDb
NB.
NB.   trg=. 'c:/smugmirror/documents/xrefdb/mirror.db'
NB.   gpx=. gpxfrrecent trg
NB.   (toHOST gpx) write 'c:/pd/coords/gpx/recent geotagged images.gpx'
NB.
NB. dyad: clGpx =. clGpxFile gpxfrrecent clMirrorDb
NB.
NB.   lastgpx=. 'c:/pd/coords/gpx/geotagged test images.gpx'
NB.   lastgpx gpxfrrecent trg

MIRRORGPXFILE gpxfrrecent y
:
waydate=. waystamp gpx=. read x NB. extract last waypoint date

NB. the last upload date is shifted forward to partly compensate
NB. for the mixture of UTC and local dates. The times in the database
NB. come from many time zones and many timestamps are just approximations.
sql=. GpxGeotaggedMirror_sql , ' and UploadDate > date("",waydate,"", ''+16 hours'') order by UploadDate d
>...>esc '
fmtmirrorgpx sql fsd y
```

```
)

gpxtimestamp=: 3 : 0

NB. *gpxtimestamp v-- format time for Garmin gpx as: yyyy-mm-ddThr:mn:scZ
NB.
NB. monad: cl =. gpxtimestamp nlTime | ntTime
NB.
NB.   gpxtimestamp 6!:0 ''
NB.
NB.   gpxtimestamp 10 # ,: 6!:0 '' NB. table

r=. }: $y
t=. _6 [\ , 6 {"1 y
d=. '--T::' 4 7 10 13 16 }"1 [ 4 3 3 3 3 3 ": <.t
c=. {: $d
d=. ,d
d=. '0' (I. d=' ')} d
'Z' ,"1~ (r,c) $ d
)

NB. extract html placeholder variable names
htmlvarbs=: { -.&.> (<{'{'})"_

NB. file name from fully qualified file names
justfile=: (] #~ [: -. [: +./\ '.'&=)@ (] #~ [: -. [: +./\ e.&':\')
```

```
parsecsv=: 3 : 0
```

```
NB.*parsecsv v-- parses comma delimited files. (x) is the field
```

```
NB. delimiter. Lines are delimited with either CRLF or LF
```

```
NB.
```

```
NB. monad: btcl =. parsecsv cl
```

```
NB. dyad: btcl =. ca parsecsv cl
```

```
NB.
```

```
NB. ', ' parsecsv read 'c:\comma\delimited\text.csv'
```

```
', ' parsecsv y
```

```
:
```

```
'separator cannot be the " character' assert -. x -: ''
```

```
NB. CRLF delimited *.csv text to char table
```

```
y=. x ,. ];._2 y -. CR
```

```
NB. bit mask of unquoted " field delimiters
```

```
b=. -. }. ~:/\ ''' e.~ ' ' , , y
```

```
b=. ($y) $ b *. , x = y
```

```
NB. use masks to cut lines
```

```
b <;._1"1 y
```

```
)
```

```
patpartstr=: 4 : 0
```

```
NB.*patpartstr v-- split list into sublists of pattern and non-pattern.
NB.
NB. dyad: (ilIdx ;< blcl) =. clPattern patpartstr clStr
NB.
NB. 'hoo' patpartstr 'hoohooohoo'
NB. 'ab.c' patpartstr 'abhc yada yada abNcabuc boo freaking hoo'
NB. 'nada' patpartstr 'nothing to match'
NB.
NB. NB. result pattern indexes and split list
NB. 'idx subtrs'=. 'yo[a-z]*' patpartstr 'yo yohomeboy no no yoman'
NB. idx{subtrs NB. patterns

NB. require 'regex' !(*)=. rxmatches
if. #pat=. , "2 x rxmatches y do.
  mask=. (#y)#0
  starts=. 0 {"1 pat
  ends=. starts + <: 1 {"1 pat
  m1=. 1 (0,starts)} mask
  m2=. _1 (|.!. 0) 1 ends} mask
  m2=. m1 +. m2
  mask=. 1 starts} mask
  idx=. (m2 {.;.1 mask) # i. +/m2
  idx;< m2 <;.1 y
else.
  (i.0);<<y
end.
)
```

*NB. reads a file as a list of bytes*

```
read=: 1!:1&([]`<@.(32&>@{3!:0)))
```

*NB. xml BEGIN and END tags*

```
tags=: '<'&,@,&'>' ; '</'&,@,&'>'
```

```
timestamp=: 3 : 0
```

*NB.\*timestamp v-- formats timestamp as dd mmm yyyy hr:mn:sc*

*NB.*

*NB. monad: cl =. timestamp zu / nlTime*

*NB.*

*NB. timestamp '' NB. empty now*

*NB. timestamp 2007 9 16 NB. fills missing*

*NB. timestamp 1953 7 2 12 33*

```
if. 0 = #y do. w=. 6!:0'' else. w=. y end.
```

```
r=. }: $ w
```

```
t=. 2 1 0 3 4 5 {"1 [ _6 [\ , 6 {"1 <. w
```

```
d=. '+++::' 2 6 11 14 17 }"1 [ 2 4 5 3 3 3 ": t
```

```
meth=. _3[\ ' JanFebMarAprMayJunJulAugSepOctNovDec '
```

```
d=. ,((1 {"1 t) { meth) 3 4 5 }"1 d
```

```
d=. '0' (I. d=' ') } d
```

```
d=. ' ' (I. d='+') } d
```

```
(r,20) $ d
```

```
)
```

*NB. appends trailing line feed character if necessary*

```
tlf=: ] , ((10{a.})"_ = {:) }. (10{a.})"
```

*NB. extract waypoint date from gpx metadata header*

```
waystamp=: [: alltrim '=' afterlaststr '</text>' beforestr ]
```

*NB. writes a list of bytes to file*

```
write=: 1!:2 ]`<@.(32&>@{3!:0))
```

*NB.POST\_gpxutils post processor.*

```
smoutput IFACE=: (0 : 0)
```

```
NB. (gpxutils) interface word(s): 20220618j183913
```

```
NB. -----
```

```
NB. allrecent      NB. all recent images from last waypoint generation
```

```
NB. csvfrwpt      NB. poi CSV text from waypoint text file
```

```
NB. gpxfrmapkml   NB. gpx from Google maps kml
```

```
NB. gpxfrmirror   NB. extracts geotagged images from mirror_db and generates gpx
```

```
NB. gpxfrpoicsv   NB. converts poi csv files to gpx
```

```
NB. gpxfrrecent   NB. gpx from recent waypoints
```

```
)
```

```
cocurrent 'base'
```

```
coinert 'gpxutils'
```



## Index

afterlaststr, 7  
afterstr, 7  
AllMirror\_sql, 5  
allrecent, 7  
alltrim, 8  
assert, 8  
  
beforestr, 8  
betweenstrs, 8  
boxopen, 9  
  
changestr, 9  
CR, 5  
csvfrwpt, 10  
ctl, 11  
  
dblquote, 11  
  
eletags, 11  
  
fmtmirrorgpx, 12

fsd, 13  
fst, 14  
  
geteletext, 14  
GPXFRKMLHEADER, 4  
gpxfrmapkml, 14  
gpxfirmirror, 16  
gpxfrpoicsv, 17  
gpxfirrecent, 19  
GpxGeotaggedMirror\_sql, 6  
GPXHEADER, 4  
GPXNAMECHARS, 6  
GPXSMUGPLACEMARK, 5  
gpxtimestamp, 20  
GPXTRAILER, 5  
  
HTMLVARBPATTERN, 6  
htmlvarbs, 20  
  
IFACE, 24

IFACEWORDSgpxutils, 6  
  
justfile, 20  
  
LF, 6  
  
MIRRORGPXFILE, 6  
  
parsecsv, 21  
patpartstr, 21  
  
read, 23  
ROOTWORDSgpxutils, 6  
  
tags, 23  
timestamp, 23  
tlf, 24  
  
VMDgpxutils, 6  
  
waystmp, 24  
write, 24