

GrabURL

Serge Emond

Copyright © Copyright1996-97 by Serge Emond

COLLABORATORS

	<i>TITLE :</i> GrabURL		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Serge Emond	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	GrabURL	1
1.1	GrabURL.guide/Contents	1
1.2	GrabURL.guide/About English	2
1.3	GrabURL.guide/Introduction	2
1.4	GrabURL.guide/Features	2
1.5	GrabURL.guide/Legal information	2
1.6	GrabURL.guide/License	2
1.7	GrabURL.guide/Legal issues/No warranty	3
1.8	GrabURL.guide/Requirements	3
1.9	GrabURL.guide/History	4
1.10	GrabURL.guide/What's next	4
1.11	GrabURL.guide/Bugs	5
1.12	GrabURL.guide/Credits	5
1.13	GrabURL.guide/Author	5
1.14	GrabURL.guide/Flags	5
1.15	GrabURL.guide/This is BETA	5
1.16	GrabURL.guide/Authentication	6
1.17	GrabURL.guide/Arguments	6
1.18	GrabURL.guide/Arguments/URL	7
1.19	GrabURL.guide/Arguments/MaxCount	7
1.20	GrabURL.guide/Arguments/MaxTime	8
1.21	GrabURL.guide/Arguments/MaxBytes	8
1.22	GrabURL.guide/Arguments/NoBG	8
1.23	GrabURL.guide/Arguments/NoHRef	9
1.24	GrabURL.guide/Arguments/NoSrc	9
1.25	GrabURL.guide/Arguments/Pattern	9
1.26	GrabURL.guide/Arguments/SaveRoot	9
1.27	GrabURL.guide/Arguments/WorkFile	10
1.28	GrabURL.guide/Arguments/InFile	10
1.29	GrabURL.guide/Arguments/Help	10

1.30	GrabURL.guide/Arguments/HeaderOnly	10
1.31	GrabURL.guide/Arguments/NoDirs	10
1.32	GrabURL.guide/Arguments/NotExists	10
1.33	GrabURL.guide/Arguments/Query	11
1.34	GrabURL.guide/Arguments/Recursive	11
1.35	GrabURL.guide/Arguments/Retry	11
1.36	GrabURL.guide/Arguments/SaveHeaders	11
1.37	GrabURL.guide/Arguments/Delay	11
1.38	GrabURL.guide/Arguments/Depth	12
1.39	GrabURL.guide/Arguments/MaxSize	12
1.40	GrabURL.guide/Arguments/MinSpaceLeft	12
1.41	GrabURL.guide/Arguments/IfModified	12
1.42	GrabURL.guide/Defaults	12
1.43	GrabURL.guide/MIME Types	15
1.44	GrabURL.guide/Browsing Anonymously	16

Chapter 1

GrabURL

1.1 GrabURL.guide/Contents

GrabURL

Version 1.02

Copyright (c) 1996-97 Serge Emond

[Important note about this English documentation](#)

[Introduction](#)

[Features](#)

[Legal information](#)

[Requirements](#)

[This is a beta release](#)

[Usage](#)

[Flags](#)

[Setting the defaults](#)

[Authentication](#)

[Browsing Anonymously](#)

[Known bugs](#)

[The future](#)

[Credits](#)

[Author](#)

[Program history](#)

1.2 GrabURL.guide/About English

A word or two about English

My primary language is French. I read English very well but I'm sure what I write is awful. Sorry about that.

I could write this in French but maybe you could not read it!

When I will have some free time I will write the docs in French too but that's not now!

1.3 GrabURL.guide/Introduction

Introduction

GrabURL is an HTTP robot.

An HTTP robot is a program that can automatically download one or more files using HTTP. Then it can scan those files to search for references to other files that it download upon some conditions and so on...

GrabURL is written in ARexx for a maximum flexibility. It was made to fit my needs.

You can create your own scripts using GrabHTTP, UrlManager and ScanHTML, simply consider GrabURL as a demo script.

1.4 GrabURL.guide/Features

Features

- Use 3 'C' programs, which means QUICK compared with ARexx-only scripts
- Is ARexx, which means flexible compared with 'C' only programs.
- Give a lot of control to the user
- Puts the full URL in file's comment
- Just read this document to see what GrabURL has to offer.

1.5 GrabURL.guide/Legal information

Legal information

License

No warranty

1.6 GrabURL.guide/License

License

GrabURL is released under the concept of freeware. This means you are allowed to use and copy this program freely, as long as the following requirements are fulfilled:

All files are copied without any alterations or modifications. If any extra files are added, it must be obvious that they do not belong to the original distribution, and that they do not need to be included in any redistribution. Exception: So called "BBS ads" may not be added.

The copying is done on a non-commercial basis. A small fee to cover media costs etc. may be charged.

The copier is not claiming the copyright of this program.

You are allowed to modify GrabURL.rexx and redistribute it but not to add it to the original distribution.

Any exceptions from the above require a written permission from the author.

1.7 GrabURL.guide/Legal issues/No warranty

No warranty

There is no warranty for the programs, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holder and/or other parties provide the programs "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the programs is with you. Should the programs prove defective, you assume the cost of all necessary servicing, repair or correction.

In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may redistribute the programs as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the programs (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the programs to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

1.8 GrabURL.guide/Requirements

Requirements

GrabURL requires GrabHTTP, UrlManager and ScanHTML. You should have received those with this script.

GrabHTTP requires AmigaOS v39+ & AmiTCP 4.0+.

GrabURL requires rexxdossupport.library 2.0+. (Can be found on AmiNet)

1.9 GrabURL.guide/History

The Past

Version 0 Revision 1 - 19.8.1996 and before

Nothing recorded.

Version 0 Revision 2 - 20.8.1996

- Added **MaxCount**
- Added **MaxTime**
- Added **MaxBytes**
- Added **NoSrc**
- Added **NoBG**
- Added **NoHRef**

Version 0 Revision 3 - 8.9.1996

- Added filecomment support
- Added config opts to change the font used by GrabHTTP.
- Other stuff not recorded.

Version 0 Revision 4 - 7.12.1996

- Added def.WriteBufSize.

Version 0 Revision 5 - 14.12.1996

- Added def.Translate.

Version 0 Revision 6 - 16.12.1996

- Added def.TranslateTo and changed the way the translation works.

Version 0 Revision 7 - 6.1.1997

- Minor corrections (Some " added, removed "*" from translate default, ...)
- All "GrabHTTP", "UrlManager" & "ScanHTML" words have been replaced by a link to the corresponding guides. (Thanks to Tom Byrer)
- Added def.FollowMoved, def.FollowTMoved, def.TouchOldDirs, def.Progress & def.ExitOnClose

14.1.1997 - 0.7 -> 1.0: Works well (at least here!).

Version 1.01 - 17.1.1997

- Authentication support

Version 1.02 - 18.1.1997

- Sets a minimal stack to prevent crashes

1.10 GrabURL.guide/What's next

What's Next - The To Do List

- Add FTP support via NCFTP or FTPMount (or GrabFTP? :)

1.11 GrabURL.guide/Bugs

Known Bugs

- Urls beginning with "ftp://" are always flagged as 'Q'ueued

1.12 GrabURL.guide/Credits

Credits

GrabURL was entirely written by [Serge Emond](#) although GetURL.rexx inspired me for the list of options. GetURL was written by James Burton and can be found on AmiNet (comm/www/GetUrl-1.06.lha).

1.13 GrabURL.guide/Author

About the author

Name: Emond

First Name: Serge

Current email: emonds@jsp.umontreal.ca

Current web: <http://www.jsp.umontreal.ca/~emonds/>

1.14 GrabURL.guide/Flags

Flags

Each url maintained in memory have a "flag" representing its status. Here is a list of the flags used by GrabURL:

Q Queued. This file is to be received.

R Received. This file has been received.

F Failed. An error occured while receiving.

X Error. (ie url inexistent)

U Unused/Unknown. This cannot be processed. (Ie do not begins with "http://")

For more information about flags, see:

[WorkFile](#)

1.15 GrabURL.guide/This is BETA

This is a BETA release

This release is BETA. This means it has not been fully tested. I release it only because it seems stable and it works well on my computer. As stated in [legal issues](#) I may not be responsible for any damage caused by the use of this program.

1.16 GrabURL.guide/Authentication

Authentication

Authentication is used by some servers to restrict files to registered users. (Ie paying websites, development sites, ...) GrabURL supports the "Basic" authentication which is the only HTTP 1.0 official scheme.

The implementation is simple: when a server returns the error "401 Unauthorized", it also send what is called a "realm": a string associated with a specific web site (a site may have multiple realms).

Upon reception of such an error, GrabURL looks in its database and send the appropriate username and password.

If you are registered on some sites then you should tell GrabURL about your informations so you can download protected files.

In order to do this, you have to create a file somewhere on your harddrive and put the name in the "def.PasswordFile" field. (See the [Setting the defaults](#) section)

In this file, lines beginning with ";" are comments. There should be one realm/line following this format:

```
realm:userid:password
```

All information is case-sensitive and spaces means something... Don't add extra spaces!!

Here's an example file:

```
;Mumblio Programming
```

```
Mumblio Prg:serge:molybden
```

```
;SpiderWeb
```

```
Spider Corp:serge:771xD
```

In this file, I'm registered on the realm "Mumblio Prg", with the name "serge" and the password "molybden". Being a registered user for the fictive web browser SpiderWeb, I can download new versions with the same user name and the password "771xD".

If you don't know the realm of one or more places, simply tell GrabURL to download a protected file: it will tell you what is the realm.

For example, if you want to know the realm used on `http://www.fictif.net/private/`, just execute this:

```
GrabURL http://www.fictif.net/private/
```

You should get something like this:

```
[ 1] http://www.fictif.net/private/... [Auth] Unknown realm: "Fictif Mumbo"
```

which tells you that the realm is "Fictif Mumbo".

1.17 GrabURL.guide/Arguments

Arguments

GrabURL can only be used from a Shell.

You can stop it by sending CTRL-C. If a file is being transferred, you should stop GrabHTTP with the close gadget (or wait until the transfert is over).

Text arguments

URL Url to download.

PATTERN Pattern to select urls.

INFILE File containing an url list.

SAVEROOT Directory where to put urls.

WORKFILE File containing all urls with their flags.

Numeric arguments

DELAY Delay to wait between each grab.

DEPTH Recursion level.

MAXCOUNT Max. number of file to grab.

MAXTIME Max. time to grab.

MAXSIZE Max. size a file can have.

MINSPACELEFT Min. space to leave on disk.

Switches

HELP Display arguments.

HEADERSONLY Save the headers but not the files.

IFMODIFIED Grab the file if modified since last grab.

NOBG Don't grab background images.

NODIRS Don't create directories - save in current dir.

NOHREF Don't grab referenced files.

NOSRC Don't grab src urls.

NOTEXISTS Grab only if file does not exists on disk.

QUERY Allow '?' in urls.

RECURSIVE Allow recursion.

RETRY Retry urls that failed.

SAVEHEADERS Save transfer headers to disk.

1.18 GrabURL.guide/Arguments/URL

URL (URL/M)

This allows you to add url(s) directly from the command line. These URLs will be flagged as **Queued** . You can add multiple urls this way. Any url added on the command line will prevail over file-urls. (If an url flagged received in the **workfile** will be flagged not received).

An url must begins with "HTTP://" since this is the only protocol supported for now.

See also:

InFile

WorkFile

1.19 GrabURL.guide/Arguments/MaxCount

MaxCount (MAXCOUNT=MC/N) (Abbrev: MC)

This is the maximal number of file to download. You can use this switch with **WorkFile** to download only a defined number of file per day. Use 0 disable this option. Default = 0.

See also:

MaxTime

MaxBytes

1.20 GrabURL.guide/Arguments/MaxTime

MaxTime (MAXTIME=MT/N) (Abbrev: MT)

This is the maximal time in minutes to download. You can use this switch with **WorkFile** to download only a certain time / day. Use 0 to disable this option. Default = 0.

This is not exact. For example, if you specify MaxTime=10 and you download this:

Bozo.jpg 3 mins (Tot: 3 mins)

Clown.jpg 6 mins (Tot: 9 mins)

Nuage.png 2 mins (Tot: 11 mins)

<it stops here>

See also:

MaxBytes

MaxCount

1.21 GrabURL.guide/Arguments/MaxBytes

MaxBytes (MAXBYTES=MB/N) (Abbrev: MB)

This is the maximal number of bytes to download. You can use this switch with **WorkFile** to download only a certain quantity of bytes per day. Use 0 to disable this option. Default = 0.

This is not an exact number. For example, if you specify MaxBytes=4096 and you download this:

Bozo.jpg 1000 bytes (Tot: 1000 bytes)

Clown.jpg 3090 bytes (Tot: 4090 bytes)

Nuage.png 1500 bytes (Tot: 5590 bytes)

<it stops here>

See also:

MaxCount

MaxTime

1.22 GrabURL.guide/Arguments/NoBG

NoBG (NOBG/S)

If specified, no background image will be grabbed. This is only useful in conjunction with **Recurvise** .

See also:

Recursive

NoHRef

NoSrc

1.23 GrabURL.guide/Arguments/NoHRef

NoHRef (NOHREF/S)

If specified, no file referenced will be grabbed. This is only useful in conjunction with [Recursive](#) .

An HREF file is a file on which you have to click to get it when you are in a web browser. (Images, another web page, etc...)

See also:

[Recursive](#)

[NoBG](#)

[NoSrc](#)

1.24 GrabURL.guide/Arguments/NoSrc

NoSrc (NOSRC/S)

If specified, all urls referred with a SRC tag will not be grabbed. This is only useful in conjunction with [Recursive](#) .

SRC tags are used to specify an image to be shown with the text. It is also used with non-standard kludges like the frames of NetScape. Thus using NOSRC will prevent grabbing HTML referenced by frames AND images to be shown with the text. I'm sorry but there is no way to distinguish them in this version.

See also:

[Recursive](#)

[NoBG](#)

[NoHRef](#)

1.25 GrabURL.guide/Arguments/Pattern

Pattern (PATTERN=P/K) (Abbrev: P)

This allows you to specify an AmigaDOS pattern. Each URL recursively found will only be added if it matches this pattern.

1.26 GrabURL.guide/Arguments/SaveRoot

SaveRoot (SAVEROOT=SR/K) (Abbrev: SR)

This is the directory to use as a base for all files we receive.

Suppose we execute:

GrabURL http://www.da.com/flowers/images/coquelicot.png sr ram:

Then the file will be saved as

ram:www.da.com/flowers/images/coquelicot.png

The default root dir is set in GrabURL. (See [Setting Defaults](#))

1.27 GrabURL.guide/Arguments/WorkFile

WorkFile (WORKFILE=WF/K) (Abbrev: WF)

This is a file where all the urls processed are saved. If the file exists at run-time, it will be loaded. This way you can break a transfer and continue it later.

Each line of that file follow this template:

<flag> <depth> <url>

See also:

[Flags](#)

[Depth](#)

[InFile](#)

[URL](#)

1.28 GrabURL.guide/Arguments/InFile

InFile (INFILE=IN/K) (Abbrev: IN)

This is a file containing one url/line. These urls are processed exactly as if they were given on the command line.

See also:

[URL](#)

1.29 GrabURL.guide/Arguments/Help

Help (HELP/S) (Abbrev: H)

This gives the supported command line options and a brief description for each one.

1.30 GrabURL.guide/Arguments/HeaderOnly

HeaderOnly (HEADERONLY=HO/S) (Abbrev: HO)

When present, this indicates GrabURL not to download the file but only the transfer header.

This is only useful in conjontion with [SaveHeaders](#) .

1.31 GrabURL.guide/Arguments/NoDirs

NoDirs (NODIRS=ND/S) (Abbrev: ND)

This tells GrabURL not to create directory but to put the file in the current directory. See [SaveRoot](#) for more explanations.

1.32 GrabURL.guide/Arguments/NotExists

NotExists (NOTEXISTS=NE/S) (Abbrev: NE)

If given, GrabURL will only download files not already on disk.

1.33 GrabURL.guide/Arguments/Query

Query (QUERY=Q/S) (Abbrev: Q)

If present, GrabURL will download files containing '??'.

Generally those files are used to send information to the server. For example Search Engines use it, some counters use it, etc..

You should not use this switch.

1.34 GrabURL.guide/Arguments/Recursive

Recursive (RECURSIVE=R/S) (Abbrev: R)

If present, GrabURL will scan each received HTML file for new urls. These urls will then be added to the list and eventually to your disk.

See also:

[Depth](#)

[Pattern](#)

1.35 GrabURL.guide/Arguments/Retry

Retry (RETRY/S)

This tells GrabURL to retry to download files that failed the first time.

Each URLs marked Failed is only retried one time. The retry happend after all the normal URLs have been processed.

1.36 GrabURL.guide/Arguments/SaveHeaders

SaveHeaders (SAVEHEADERS=SH/S) (Abbrev: SH)

If given GrabURL will save the transfer headers along with the files.

The header file has the same name than the file itself, with ".HDR" appended.

See also:

[HeaderOnly](#)

1.37 GrabURL.guide/Arguments/Delay

Delay (DELAY/N)

This is the number of seconds to wait between each file. You should not use a delay of 0 seconds if you plan to download a LOT of files - using this switch helps to diminish a little bit the load on the net.

The factory setting is 0 second and can be changed. (See [Defaults](#))

1.38 GrabURL.guide/Arguments/Depth

Depth (DEPTH=D/K) (Abbrev: D)

This is the level of recursion to use. (ie How deep to look in the files)

Level '0' means the "current" file. All files contained in the current files have a level of "1" and so on.

Specifying 5 (along with the **Recursive** switch) would grab the files on the command line and specified with **InFile** , giving them a "depth" of 5. The files contained in them would have a depth of 4. Then 3, 2, 1 and finally 0. The files contained in the depth-0 file won't be grabbed.

The default is "-1" which means "grab the world".

1.39 GrabURL.guide/Arguments/MaxSize

MaxSize (MAXSIZE=MS/N) (Abbrev: MS)

With this you can specify the maximal size a file can have in order to download it.

Ie "MaxSize 4096" will refuse a file having 4097 bytes but accepts 4096.

0 is the default which disable this option.

1.40 GrabURL.guide/Arguments/MinSpaceLeft

MinSpaceLeft (MINSPACELEFT=MSL/N) (Abbrev: MSL)

This is the minimal space to leave free on the device when downloading a file. The default is 0 (disabled).

1.41 GrabURL.guide/Arguments/IfModified

IfModified (IFMODIFIED=IM/S) (Abbrev: IM)

If present, each file that already exists on the disk will only be grabbed if the file on the server is newer than the one on the disk.

Some servers don't care about If-Modified information with .html files. Others don't use this at all. (It's worth a try!)

See also:

[Defaults - Setting the timezone](#)

1.42 GrabURL.guide/Defaults

Changing the defaults

The defaults are set directly in GrabURL.rexx using an ASCII editor.

When in the editor (ie GoldEd, CignusEd, ...), load "GrabURL.rexx" and go down some lines. You should see a line beginning with:

DoDefaults:

You are now where the defaults are all set.

Here's a list of the defaults used for the command line **arguments** :

opts.Url.Count (Factory: 0)

Must always be 0.

opts.Depth (Factory: -1)

This is the default **depth** to use. When doing recursion, this tells GrabURL not to care about depth.

opts.InFile (Factory: "")

This is the default file to load when starting GrabURL. This file contains one url/line. See **InFile** .

opts.MaxSize (Factory: 0)

This is the maximal size a file can have in order to be grabbed. See **MaxSize** .

opts.MinSpaceLeft (Factory: 0)

This is the minimal space to leave on disk when grabbing. See **MinSpaceLeft** .

opts.Pattern (Factory: "")

This is the default pattern to use when doing recursion. See **Pattern** .

opts.SaveRoot (Factory: 'Work:Urls/')

This is the default path where to save urls. See **SaveRoot** .

opts.WorkFile (Factory: "")

This is the default **work file** to use.

opts.Delay (Factory: 0)

This is how much seconds to wait between each grabbed file. See **Delay** .

opts.MaxTime (Factory: 0)

This is the maximal time in minutes to download. See **MaxTime** .

opts.MaxCount (Factory: 0)

This is the maximal number of file to download. See **MaxCount** .

opts.MaxBytes (Factory: 0)

This is the maximal number of bytes to download. See **MaxBytes** .

Here's a list of the settings that cannot be modified on the command line:

def.KeepReceived (Factory: 1)

1 = Change the **flag** of received urls

0 = Remove them from the list

def.KeepFailed (Factory: 1)

1 = Change the **flag** of failed urls

0 = Remove urls that failed from the list.

def.KeepUnused (Factory: 1)

1 = Change the **flag** of urls that could not be processed.

0 = Remove them from the list.

def.FollowMoved (Factory: 1)

1 = Add the new URL given by the server when a "301 Moved" is received.

F = Mark it "Failed".

X = Mark "Error".

def.FollowTMoved (Factory: 1)

1 = Add the new URL given by the server when a "302 Moved Temporarily" is received.

F = Mark it "Failed".

X = Mark "Error".

def.Secure (Factory: 1)

1 = Save the workfile each time an url is modified (ie received, failed, removed, ...)

0 = Save the workfile only when GrabURL is completely done.

def.Accept (Factory: '*/*')

This is the **MIME** type(s) the remote host is allowed to send.

def.Email (Factory: '')

This is the UserName/Email address to send to each remote host we encounter. Using '' forces GrabURL to send NO user information across the net. See **Browsing anonymously** . This information can look like:

Serge Emond <emonds@jsp.umontreal.ca>

def.TimeZone (Factory: 300)

This is the number of minutes to add to the local time to obtain GMT time. It is only used when using the **IfModified** argument. The default is 5 hours (300 minutes) which is Québec/Ontario and the eastern part of USA.

def.WriteBufSize (Factory: 16*1024 (16k))

This is the size of the buffer GrabHTTP will use for the file itself. The received data will be written only every def.WriteBufSize bytes.

def.Translate (Factory: '~:[]()')

If it is not '' then the characters given will be translated to chars in def.TranslateTo when saving the file to disk. For example, if def.Translate="z!" and def.TranslateTo="_+" then all "z" in the filename will be changed to "_" and all "!" to "+". To specify "*" you must write "***". See GrabHTTP documentation.

def.TranslateTo (Factory: copies("_", length(def.Translate)))

String to translate to. See def.Translate. This string MUST be at least the length of def.Translate.

def.TouchOldDirs (Factory: 1)

If 1, the already existing directory-names will have their date set to now. (Ie http://hey.org/x/ creates "hey.org" then "hey.org/x". If this option is 1 and some or all of these dirs exists, their date will be reset.)

def.TempFile (Factory: 't:GrabURL.'UID)

This is a temporary file used when scanning received HTML files to do recursion. UID is a variable containing a unique ID and should be used in the name (as in the factory setting) to prevent collision if you run multiple GrabURL at the same time.

def.ParsePattern (Factory: '(FTP|HTTP)://#?')

Only urls matching this pattern are added to the url list. If you want to see if there was ftp urls, allow them with this setting and use a workfile. You can then see them by searching for "ftp:" in the workfile.

def.PasswordFile (Factory: '')

This is the filename of a data file containing password information. See **Authentication** for more information.

def.Path (Factory: '')

This is the path where to find ScanHTML, GrabHTTP & UrlManager executables.

def.ScanHTML (Factory: def.Path'ScanHTML')

This is the path & name of ScanHTML. See **Requirements** .

def.GrabHTTP (Factory: 'run <>nil: 'def.Path'')

This is the name of GrabHTTP. It must be started with "run". See **Requirements** .

def.UrlManager (Factory: 'run <>nil: ''def.Path'UrlManager')

This is the name of UrlManager. It must be started with "run". See **Requirements** .

def.Output (Factory: 'CONSOLE:')

This is the filename where GrabURL will output all its... output. "CONSOLE:" will output to the console from which GrabURL was started.

def.Progress (Factory: 1)

If 1, tells GrabHTTP to display progress informations in a window. If 0, there's no window telling how much of the file has been received.

def.ExitOnClose (Factory: 0)

If 0, and the user presses the CloseGadget of GrabHTTP's window then the current file is marked Failed and the grab continues. If 1, GrabURL exits.

def.Font (Factory: 'topaz.font')

This is the font used for GrabHTTP's progress window. It must be fixed-width.

def.FSize (Factory: 8)

This is the size of the font.

um (Factory: 'UM'UID)

This is the name of the port to use to communicate with UrlManager. You should use the variable UID as in the factory setting to prevent collision.

gh (Factory: 'GH'UID)

This is the name of the port to use to communicate with GrabHTTP. You should use the variable UID as in the factory setting to prevent collision.

1.43 GrabURL.guide/MIME Types

MIME

What follows come directly from AWeb's HTML documentation:

MIME (Multipurpose Internet Mail Extensions) is a mechanism for specifying and describing the format of Internet message bodies. It was primarily designed for e-mail, but MIME types are used also to identify the type of data in the HTTP protocol, the most widely used protocol on the World Wide Web.

A MIME type consists of a type and a subtype. The type describes the major class of data, like text or image. The subtype is used for a subdivision of the major type into different formats, like GIF or JPEG images.

According to RFC 1521, the following official MIME types are defined:

TEXT/HTML

This is a document in the HTML hypertext format. Virtually all pages on the Web are in this format.

TEXT/PLAIN

This type is used for plain text documents (normally in ASCII).

APPLICATION/OCTET-STREAM

This describes a binary file. The file could be processed by some application. An example of this would be an LHA archive.

APPLICATION/POSTSCRIPT

The document is in PostScript format.

IMAGE/GIF

IMAGE/JPEG

These are images, in GIF and JPEG format.

AUDIO/BASIC

This type is used for audio data encoded using 8-bit ISDN mu-law [PCM].

VIDEO/MPEG

This is an animation in MPEG format.

In addition to these official types and subtypes, it is allowed to define extension MIME types and subtypes. These should start with X- to avoid collisions with future official MIME types.

1.44 GrabURL.guide/Browsing Anonymously

Browsing Anonymously

An HTTP request can have a field containing email and name information about the user. This way, some people send their email address on every server they connect to. Most of the time it won't do anything but... there are twisted webmasters who use this information to send annoying emails ie to sell products.

If you want to see if your Web Browser sends such information without letting you knowing about it, have a look at this page: [HTTP://www.uiuc.edu/~ejk/WWW-privacy.html](http://www.uiuc.edu/~ejk/WWW-privacy.html).

By default GrabURL won't send any information about the user. In fact, there is absolutely no function allowing to grab any information from your system without your consent so you can be sure it won't send anything. :)

However you really should read about Robot Guidelines in [GrabHTTP.guide](#).

See the [Defaults](#) section to see how to set an EMail address with GrabURL.
