Httrack Users Guide (3.10)

By Fred Cohen

Background and Introduction

I started using httrack in mid-2000 and found it to be an excellent tool for imaging web sites. Various words are used to describe this process - from imaging to mirroring to snaking and so on. I will be using a variety of these words in my description.

I have used many such tools over the years, have performed many manual and semi-automatic operations of similar sorts, and written partial programs to do similar functions, but - at least for now - httrack seems to me to be the best option for this function.

The only problem I encountered when using httrack was that it is so rich with features that I could never really figure out precisely the right thing to do at any given point. I was using recepies rather than knowledge to get the job done - and I was pestering the authors for those recepies. After a few days of very helpful assistance from the authors I volenteered to write a users manual for httrack - and here it is. I hope it gets the job done.

Basics

Httrack is a program that gets information from the Internet, looks for pointers to other information, gets that information, and so forth. If you ask it to, and have enough disk space, it will try to make a copy of the whole Internet on your computer. While this may be the answer to Dilbert's boss when he asks to get a printout of the Internet for some legal document, for most of us, we want to get copies of just the right part of the Internet, and have them nicely organized for our use. This is where httrack does a great job. Here's a simple example:

```
httrack "http://www.all.net/" -O "/tmp/www.all.net" "+*.all.net/*" -v
```

In this example, we ask httrack to start the Universal Resource Locator (URL) http://www.all.net/ and store the results under the directory /tmp/www.all.net (the -O stands for "output to") while not going beyond the bounds of all the files in the www.all.net domain and printing out any error messages along the way (-v means verbose). This is the most common way that I use httrack. Please note that this particular command might take you a while - and run you out of disk space.

This sort of a mirror image is not an identical copy of the original web site - in some ways it's better such as for local use - while in other ways it may be problematic - such as for legal use. This default mirroring method changes the URLs within the web site so that the references are made relative to the location the copy is stored in. This makes it very useful for navigating through the web site on your local machine with a web browser since most things will work as you would expect them to work. In this example, URLs that point outside of the www.all.net domain space will still point there, and if you encounter one, the web browser will try to get the data from that location.

For each of the issues discussed here - and many more - httrack has options to allow you to make different choices and get different results. This is one of the great things about httrack - and one of the the real major problems with using it without the knowledge of all that it can do. If you want to know all the things httrack can do, you might try typing:

```
httrack --help
```

Unfortunately, while this outputs a though list of options, it is somewhat less helpful it might be for those who don't know what the options all mean and haven't used them before. On the other hand, this is most useful for those who already know how to use the program but don't remember some obscure option that they haven't used for some time.

The rest of this manual is dedicated to detailing what you find in the help message and providing examples - lots and lots of examples... Here is what you get (page by page - use to move to the next page in the real program) if you type 'httrack --help':

```
>httrack --help
 HTTrack version 3.03BETAo4 (compiled Jul  1 2001)
 usage:  ./httrack ] [-]
 with options listed below: (* is the default value)

General options:
  O  path for mirror/logfiles+cache (-O path_mirror[,path_cache_and_logfiles]) (--path )
 %O  top path if no path defined (-O path_mirror[,path_cache_and_logfiles])

Action options:
  w *mirror web sites (--mirror)
  W  mirror web sites, semi-automatic (asks questions) (--mirror-wizard)
  g  just get files (saved in the current directory) (--get-files)
  i  continue an interrupted mirror using the cache
  Y   mirror ALL links located in the first level pages (mirror links) (--mirrorlinks)

Proxy options:
  P  proxy use (-P proxy:port or -P user:pass@proxy:port) (--proxy )
 %f *use proxy for ftp (f0 don't use) (--httpproxy-ftp[=N])

Limits options:
  rN set the mirror depth to N (* r9999) (--depth[=N])
 %eN set the external links depth to N (* %e0) (--ext-depth[=N])
  mN maximum file length for a non-html file (--max-files[=N])
  mN,N'                  for non html (N) and html (N')
  MN maximum overall size that can be uploaded/scanned (--max-size[=N])
  EN maximum mirror time in seconds (60=1 minute, 3600=1 hour) (--max-time[=N])
  AN maximum transfer rate in bytes/seconds (1000=1kb/s max) (--max-rate[=N])
 %cN maximum number of connections/seconds (*%c10)
  GN pause transfer if N bytes reached, and wait until lock file is deleted (--max-pause[=N])

Flow control:
  cN number of multiple connections (*c8) (--sockets[=N])
  TN timeout, number of seconds after a non-responding link is shutdown (--timeout)
  RN number of retries, in case of timeout or non-fatal errors (*R1) (--retries[=N])
  JN traffic jam control, minimum transfert rate (bytes/seconds) tolerated for a link (--min-rate[=N])
  HN host is abandonned if: 0=never, 1=timeout, 2=slow, 3=timeout or slow (--host-control[=N])

Links options:
 %P *extended parsing, attempt to parse all links, even in unknown tags or Javascript (%P0 don't use) (--extended-parsing[=N])
  n  get non-html files 'near' an html file (ex: an image located outside) (--near)
  t  test all URLs (even forbidden ones) (--test)
 %L )

Build options:
  NN structure type (0 *original structure, 1+: see below) (--structure[=N])
     or user defined structure (-N "%h%p/%n%q.%t")
  LN long names (L1 *long names / L0 8-3 conversion) (--long-names[=N])
  KN keep original links (e.g. http://www.adr/link) (K0 *relative link, K absolute links, K3 absolute URI links) (--keep-links[=N])
  x  replace external html links by error pages (--replace-external)
 %x  do not include any password for external password protected websites (%x0 include) (--no-passwords)
 %q *include query string for local files (useless, for information purpose only) (%q0 don't include) (--include-query-string)
  o *generate output html file in case of error (404..) (o0 don't generate) (--generate-errors)
  X *purge old files after update (X0 keep delete) (--purge-old[=N])

Spider options:
  bN accept cookies in cookies.txt (0=do not accept,* 1=accept) (--cookies[=N])
  u  check document type if unknown (cgi,asp..) (u0 don't check, * u1 check but /, u2 check always) (--check-type[=N])
  j *parse Java Classes (j0 don't parse) (--parse-java[=N])
  sN follow robots.txt and meta robots tags (0=never,1=sometimes,* 2=always) (--robots[=N])
 %h  force HTTP/1.0 requests (reduce update features, only for old servers or proxies) (--http-10)
 %B  tolerant requests (accept bogus responses on some servers, but not standard!) (--tolerant)
 %s  update hacks: various hacks to limit re-transfers when updating (identical size, bogus response..) (--updatehack)
 %A  assume that a type (cgi,asp..) is always linked with a mime type (-%A php3=text/html) (--assume )

Browser ID:
  F  user-agent field (-F "user-agent name") (--user-agent )
 %F  footer string in Html code (-%F "Mirrored [from host %s [file %s [at %s]]]" (--footer )
 %l  preffered language (-%l "fr, en, jp, *" (--language )

Log, index, cache
  C  create/use a cache for updates and retries (C0 no cache,C1 cache is prioritary,* C2 test update before) (--cache[=N])
  k  store all files in cache (not useful if files on disk) (--store-all-in-cache)
 %n  do not re-download locally erased files (--do-not-recatch)
```

```
 %v  display on screen filenames downloaded (in realtime) (--display)
  Q  no log - quiet mode (--do-not-log)
  q  no questions - quiet mode (--quiet)
  z  log - extra infos (--extra-log)
  Z  log - debug (--debug-log)
  v  log on screen (--verbose)
  f *log in files (--file-log)
  f2 one single log file (--single-log)
  I *make an index (I0 don't make) (--index)
 %I  make an searchable index for this mirror (* %I0 don't make) (--search-index)

Expert options:
  pN priority mode: (* p3) (--priority[=N])
      0 just scan, don't save anything (for checking links)
      1 save only html files
      2 save only non html files
     *3 save all files
      7 get html files before, then treat other files
  S  stay on the same directory
  D *can only go down into subdirs
  U  can only go to upper directories
  B  can both go up&down into the directory structure
  a *stay on the same address
  d  stay on the same principal domain
  l  stay on the same TLD (eg: .com)
  e  go everywhere on the web
 %H  debug HTTP headers in logfile (--debug-headers)

Guru options: (do NOT use)
 #0  Filter test (-#0 '*.gif' 'www.bar.com/foo.gif')
 #f  Always flush log files
 #FN Maximum number of filters
 #h  Version info
 #K  Scan stdin (debug)
 #L  Maximum number of links (-#L1000000)
 #p  Display ugly progress information
 #P  Catch URL
 #R  Old FTP routines (debug)
 #T  Generate transfer ops. log every minutes
 #u  Wait time
 #Z  Generate transfer rate statictics every minutes
 #!  Execute a shell command (-#! "echo hello")

Command-line specific options:
  V execute system command after each files ($0 is the filename: -V "rm \$0") (--userdef-cmd )
 %U run the engine with another id when called as root (-%U smith) (--user )

Details: Option N
  N0 Site-structure (default)
  N1 HTML in web/, images/other files in web/images/
  N2 HTML in web/HTML, images/other in web/images
  N3 HTML in web/,  images/other in web/
  N4 HTML in web/, images/other in web/xxx, where xxx is the file extension
(all gif will be placed onto web/gif, for example)
  N5 Images/other in web/xxx and HTML in web/HTML
  N99 All files in web/, with random names (gadget !)
  N100 Site-structure, without www.domain.xxx/
  N101 Identical to N1 exept that "web" is replaced by the site's name
  N102 Identical to N2 exept that "web" is replaced by the site's name
  N103 Identical to N3 exept that "web" is replaced by the site's name
  N104 Identical to N4 exept that "web" is replaced by the site's name
  N105 Identical to N5 exept that "web" is replaced by the site's name
  N199 Identical to N99 exept that "web" is replaced by the site's name
  N1001 Identical to N1 exept that there is no "web" directory
  N1002 Identical to N2 exept that there is no "web" directory
  N1003 Identical to N3 exept that there is no "web" directory (option set for g option)
  N1004 Identical to N4 exept that there is no "web" directory
  N1005 Identical to N5 exept that there is no "web" directory
  N1099 Identical to N99 exept that there is no "web" directory
Details: User-defined option N
  %n Name of file without file type (ex: image) (--do-not-recatch)
  %N Name of file, including file type (ex: image.gif)
  %t File type (ex: gif)
  %p Path [without ending /] (ex: /someimages)
  %h Host name (ex: www.someweb.com) (--http-10)
  %M URL MD5 (128 bits, 32 ascii bytes)
  %Q query string MD5 (128 bits, 32 ascii bytes)
  %q small query string MD5 (16 bits, 4 ascii bytes) (--include-query-string)
     %s? Short name version (ex: %sN)
  %[param] param variable in query string

Shortcuts:
--mirror      <="" pre="">


 For many of you, the manual is now complete, but for
the rest of us, I will now go through this listing one item at a time
with examples... I will be here a while...
```

Syntax

httrack [-option] [+] [-]

The syntax of httrack is quite simple. You specify the URLs you wish to start the process from (), any options you might want to add ([-option], any filters specifying places you should ([+]) and should not ([-]) go, and end the command line by pressing . Httrack then goes off and does your bidding. For example:

httrack www.all.net/bob/

This will use the 'defaults' (those selections from the help page marked with '*' in the listing above) to image the web site. Specifically, the defauls are:

```
  w *mirror web sites
 %f *use proxy for ftp (f0 don't use)
```

```
 cN number of multiple connections (*c8)
 RN number of retries, in case of timeout or non-fatal errors (*R1)
%P *extended parsing, attempt to parse all links, even in unknown tags or Javascript (%P0 don't use)
 NN  name conversion type (0 *original structure, 1+: see below)
 LN  long names (L1 *long names / L0 8-3 conversion)
 K   keep original links (e.g. http://www.adr/link) (K0 *relative link)
 o *generate output html file in case of error (404..) (o0 don't generate)
 X *purge old files after update (X0 keep delete)
 bN  accept cookies in cookies.txt (0=do not accept,* 1=accept)
 u check document type if unknown (cgi,asp..) (u0 don't check, * u1 check but /, u2 check always)
 j *parse Java Classes (j0 don't parse)
 sN  follow robots.txt and meta robots tags (0=never,1=sometimes,* 2=always)
 C  create/use a cache for updates and retries (C0 no cache,C1 cache is prioritary,* C2 test update before)
 f *log file mode
 I *make an index (I0 don't make)
 pN priority mode: (* p3)  *3 save all files
 D  *can only go down into subdirs
 a  *stay on the same address
 --mirror       *make a mirror of site(s) (default)
```

Here's what all of that means:

w *mirror web sites

Automatically go though each URL you download and look for links to other URLs inside it, dowloading them as well.

%f *use proxy for ftp (f0 don't use)

If there are and links to ftp URLs (URLs using the file transfer protocol (FTP) rather than the hypertext transfer protocol HTTP), go through an ftp proxy server to get them.

cN number of multiple connections (*c8)

Use up to 8 simultaneous downloads so that at any gioven time, up to 8 URLs may be underway.

RN number of retries, in case of timeout or non-fatal errors (*R1)

Retry once if anything goes wrong with a download.

%P *extended parsing, attempt to parse all links, even in unknown tags or Javascript (%P0 don't use)

Try to parse all URLs - even if they are in Javascript, Java, tags of unknown types, or anywhere else the program can find things.

NN name conversion type (0 *original structure, 1+: see below)

Use the original directory and file structure of the web site in your mirror image of the site.

LN long names (L1 *long names / L0 8-3 conversion)

If filenames do not follow the old DOS conventions, store them with the same names used on the web site.

K   keep original links (e.g. http://www.adr/link) (K0 *relative link)

Use relative rather than the original links so that URLs within this web site are adjusted to point to the files in the mirror.

o *generate output html file in case of error (404..) (o0 don't generate)

IF there are errors in downloading, create a file that indicates that the URL was not found. This makes browsing go a lot smoother.

X *purge old files after update (X0 keep delete)

Files not found on the web site that were previously there get deleted so that you have an accurate snapshot of the site as it is today - losing historical data.

bN accept cookies in cookies.txt (0=do not accept,* 1=accept)

Accept all cokkies sent to you and return them if requested. This is required for many sites to function. These cookies are only kept relative to the specific site, so you don't have to worry about your browser retaining them.

u check document type if unknown (cgi,asp..) (u0 don't check, * u1 check but /, u2 check always)

This causes different document types to be analyzed differently.

j *parse Java Classes (j0 don't parse)

This causes Java class files to be parsed looking for URLs.

sN follow robots.txt and meta robots tags (0=never,1=sometimes,* 2=always)

This tells the program to follow the wishes of the site owner with respect to limiting where robots like this one search.

C create/use a cache for updates and retries (C0 no cache,C1 cache is prioritary,* C2 test update before)

If you are downloading a site you have a previous copy of, supplemental parameters are transmitted to the server, for example the 'If-Modified-Since:' field will be used to see if files are newer than the last copy you have. If they are newer, they will be downloaded, otherwise, they will not.

f *log file mode

This retains a detailed log of any important events that took place.

I *make an index (I0 don't make)

This makes a top-level index.html file so that if you image a set of sites, you can have one place to start reviewing the set of sites.

pN priority mode: (* p3) *3 save all files

This will cause all downloaded files to be saved.

D *can only go down into subdirs

This prevents the program from going to higher level directories than the initial subdirectory, but allows lower-level subdirectories of the starting directory to be investigated.

a *stay on the same address

This indicates that only the web site(s) where the search started are to be collected. Other sites they point to are not to be imaged.

--mirror *make a mirror of site(s) (default)

This indicates that the program should try to make a copy of the site as well as it can.

Now that's a lot of options for the default - but of course there are a lot more options to go. For the most part, the rest of the options represent variations on these themes. For example, instead of saving all files, we might only want to save html files, or instead of 8 simultaneous sessions, we might want only 4.

If we wanted to make one of these changes, we would specify the option on the command line. For example:

httrack www.all.net/bob/ -c4 -B

This would restrict httrack to only use 4 siumultaneous sessions but allow it to go up the directory structure (for example to www.all.net/joe/) as well as down it (for example to www.all.net/bob/deeper/).

You can add a lot of options to a command line!

A Thorough Going Over

Now that you have an introduction, it's time for a more though coverage. This is where I go through each of the options

and describe it in detail with examples... Actually, I won't quite do that. But I will get close.

Options tend to come in groups. Each group tends to be interrelated, so it's easier and more useful to go through them a group at a time with some baseline project in mind. In my case, the project is to collect all of the information on the Internet about some given subject. We will assume that, through a previous process, I have gotten a list of URLs of interest to me. Typically there will be hundreds of these URLs, and they will be a mixed bag of sites that are full of desired information, pages with lists of pointers to other sites, URLs of portions of a web site that are of interest (like Bob's home pages and subdirectories), and so forth. Let us say that for today we are looking for the definitive colleciton of Internet information on shoe sizes from around the world.

General Options

General options:
O path for mirror/logfiles+cache (-O path_mirror[,path_cache_and_logfiles])

For this project, I will want to keep all of the information I gather in one place, so I will specify that output area of the project as /tmp/shoesizes by adding '***-O /tmp/shoesizes***' to every command line I use.. for example:

httrack http://www.shoesizes.com -O /tmp/shoesizes

The action options tell httrack how to operate at the larger level.

Action Options

Action options:
w *mirror web sites
W mirror web sites, semi-automatic (asks questions)
g just get files (saved in the current directory)
i continue an interrupted mirror using the cache
Y mirror ALL links located in the first level pages (mirror links)

If I want httrack to ask me questions - such as what options to use, what sites to mirror, etc. I can tell it to ask these questions as follows:

httrack http://www.shoesizes.com -O /tmp/shoesizes -W

I can also do:

httrack

OR

httrack -W

OR

httrack -w

The '-W' options asks whether the or not a site has to be mirrored, while the '-w' option does not ask this question but asks the remainder of the questions required to mirror the site.

The -g option allows you to get the files exactly as they are and store them in the currant directory. This is handy for a relatively small collection of information where organization isn't important. With this option, the html files will not even be parsed to look for other URLs. This option is useful for getting isolated files (e.g., httrack -g www.mydrivers.com/drivers/windrv32.exe).

If I start a collection process and it fails for ome reason or another - such as me interrupting it because I am running out of disk space - or a network outage - then I can restart the process by using the -i option:

httrack http://www.shoesizes.com -O /tmp/shoesizes -i

Finally, I can mirror all links in the first level pages of the URLs I specify. A good example of where to use whis would be in a case where I have a page that points to a lot of other sites and I want to get the initial information on those sites before mirroring them:

httrack http://www.shoesizes.com/othersites.html -O /tmp/shoesizes -Y

Proxy Options

Many users use a proxy for many of their functions.
This is a key component in many firewalls, but it is also commonly used
for anonymizing access and for exploiting higher speed communications at
a remote server.

Proxy options:
P proxy use (-P proxy:port or -P user:pass@proxy:port)
%f *use proxy for ftp (f0 don't use)

If you are using a standard proxy that doesn't require
a user ID and password, you would do something like this:

httrack http://www.shoesizes.com -O /tmp/shoesizes -P proxy.www.all.net:8080

In this case, we have asusmed that proxy.www.all.net is
the host that does the proxy service and that it uses port 8080 for this
service. In some cases you will have to ask your network or firewall
administrator for these details, however, in most cases they should be
the same as the options used in your web browser.

In some cases, a user ID and password are required for
the proxy server. This is common in corporate environments where only
authorized users may access the Internet.

httrack http://www.shoesizes.com -O /tmp/shoesizes -P fc:password@proxy.www.all.net:8080

In this case, the user ID 'fc' and the password
'password' are used on proxy.www.all.net port 8080. Again, your network or
firewall administrator can be most helpful in addressing the specifics
for your environment.

FTP normally operates through a proxy server, but for systems
that have direct connections to the Internet, the following option should help:

httrack ftp://ftp.shoesizes.com -O /tmp/shoesizes -%f0

Limits Options

Limits options:
rN set the mirror depth to N
mN maximum file length for a non-html file
mN,N' for non html (N) and html (N')
MN maximum overall size that can be uploaded/scanned
EN maximum mirror time in seconds (60=1 minute, 3600=1 hour)
AN maximum transfer rate in bytes/seconds (1000=1kb/s max)
GN pause transfer if N bytes reached, and wait until lock file is deleted
%eN set the external links depth to N (* %e0) (--ext-depth[=N])
%cN maximum number of connections/seconds (*%c10)

Setting limits provides the means by which you can
avoid running out of disk space, CPU time, and so forth. This may be
particularly helpful for those who accidentally try to image the whole
Internet.

httrack http://www.shoesizes.com -O /tmp/shoesizes -r50

In this example, we limit the directlry depth to 50
levels deep. As a general rule, web sites don't go much deeper than 20
levels or so, and if you think about it, if there are only 2
subdirectories per directory level, a directory structure 50 deep would
have about 10 trillion directories. Of course many sites have a small
number of files many levels deep in a directory structure for various
reasons. In some cases, a symbolic link will cause an infinite
recursion of directory levels as well, so placing a limit may be
advisable.

httrack http://www.shoesizes.com -O /tmp/shoesizes -m50000000

This example sets the maximum file length for non-HTML
files to 50 megabytes. This is not an unusual length for things like
tar files, and in some cases - for example when there are images of
CD-ROMs to fetch from sites, you might want a limit more like 750
megabytes.

httrack http://www.shoesizes.com -O /tmp/shoesizes -m50000000,100000

In this example, we have set a limit for html files
as well - at 100,000 bytes. HTML files are rarely larger than this,
however, in some cases larger sizes may be needed.

httrack http://www.shoesizes.com -O /tmp/shoesizes -M1000000000

This option sets the maximum total size - in bytes -
that can be uploaded from a site - in this case to 1 gigabyte.
Depending on how much disk space you have, such an option may be
worthwhile.

httrack http://www.shoesizes.com -O /tmp/shoesizes -E3600

This sets the maximum runtime for the download
process. Of course depending on the speed of your connection it may
take longer or shorter runtimes to get the same job done, and network
traffic is also a factor. 3600 seconds corresponds to one hour.

httrack http://www.shoesizes.com -O /tmp/shoesizes A100000000

This option specifies the largest number of bytes per
second that should be used for transfers. For example, you might want
to go slow for some servers that are heavily loaded in the middle of the
day, or to download slowly so that the servers at the other end are less
likely to identify you as mirroring their site. The setting above
limits my bandwidth to 100 million bytes per second - slow I know, but I
wouldn't want to stress the rest of the Internet.

httrack http://www.shoesizes.com -O /tmp/shoesizes -G100000000

In this case, the G option is used to 'pause' a
download after the first gigabyte is downloaded pending manual removal
of the lockfile. This is handy of you want to download some portion of
the data, move it to secondary storage, and then continue - or if you
want to only download overnight and want to stop before daylight and
continue the next evening. You could even combine this option with a
cron job to remove the lock file so that the job automatically restarts
at 7PM every night and gets another gigabyte.

httrack http://www.shoesizes.com -O /tmp/shoesizes %e5

In this case, httrack will only go to depth 5 for external links,
thus not imaging the entire web, but only yhose links within 5 links of these web pages.

Also note that the interaction of these options may
cause unintended consequences. For example, limiting bandwidth and
download time conspire to limit the total amount of data that can
be downloaded.

Flow Control Options

Flow control:
- ***cN number of multiple connections (*c8)***
- ***%cN maximum number of connections/seconds (*%c10)***
- ***TN timeout, number of seconds after a non-responding link is shutdown***
- ***RN number of retries, in case of timeout or non-fatal errors (*R1)***
- ***JN traffic jam control, minimum transfert rate (bytes/seconds) tolerated for a link***
- ***HN host is abandonned if: 0=never, 1=timeout, 2=slow, 3=timeout or slow***

This example allows up to 128 simultaneous downloads.
Note that this is likely to crash remote web servers - or at least fail
to download many of the files - because of limits on the number of
simultaneous sessions at many sites. At busy times of day, you might
want to lower this to 1 or 2, especially at sites that limit the number
of simultaneous users. Otherwise you will not get all of the downloads.

httrack http://www.shoesizes.com -O /tmp/shoesizes -c128

Many operating systems have a limit of 64 file
handles, including internet connections and all other files that can be
opened. Therefore, in many cases, more that 48 connections might cause
a "socket error" because the OS can not handle that many sockets. This
is also true for many servers. As an example, a test with 48 sockets on
a cgi-based web server (Pentium 166,80Meg RAM) overloaded the machine
and stopped other services from running correctly. Some servers will
ban users that try to brutally download the website. 8 sockets is
generally good, but when I'm getting large files (e.g., from a a site
with large graphical images) 1 or 2 sockets is a better selection. Here
are some other figures from one sample set of runs:

> Tests: on a 10/100Mbps network, 30MB website, 99 files (70 images (some are
> little, other are big (few MB)), 23 HTML)
> With 8 sockets: 1,24MB/s
> With 48 sockets: 1,30MB/s
> With 128 sockets: 0,93MB/s

The timeout option causes downloads to time out after
a non-response from a download attempt. 30 seconds is pretty reasonable
for many sites. You might want to increase the number of retries as
well so that you try again and again after such timeouts.

httrack http://www.shoesizes.com -O /tmp/shoesizes -%c20

This limits the number of connections per second. It is similar to the above option but allows the pace to be controlled rather than the simultanaety. It is particulsrly useful for long-term pulls at low rates that allow little impact on remote infrastructure. The default is 10 connections per second.

httrack http://www.shoesizes.com -O /tmp/shoesizes -T30

This example increases the number of retries to 5. This means that if a download fails 5 times, httrack will give up on it. For relatively unreliable sites - or for busy times of day, this number should be higher.

httrack http://www.shoesizes.com -O /tmp/shoesizes -R5

This is an interesting option. It says that in a traffic jam - where downloads are excessively slow - we might decide to back off the download. In this case, we have limited downloads to stop bothering once we reach 10 bytes per second.

httrack http://www.shoesizes.com -O /tmp/shoesizes -J10

These three options will cause the download from a host to be abandoned if (respectively) (0) never, (1) a timeout is reached, (2) slow traffic is detected, (or) (3) a timeout is reached OR slow traffic is detected.

httrack http://www.shoesizes.com -O /tmp/shoesizes -H0
httrack http://www.shoesizes.com -O /tmp/shoesizes -H1
httrack http://www.shoesizes.com -O /tmp/shoesizes -H2
httrack http://www.shoesizes.com -O /tmp/shoesizes -H3

Of course these options can be combined to provide a powerful set of criteria for when to continue a download and when to give it up, how hard to push other sites. and how much to stress infrastructures.

Link Following Options

Links options:
%P *extended parsing, attempt to parse all links, even in unknown tags or Javascript (%P0 don't use)
n get non-html files 'near' an html file (ex: an image located outside)
t test all URLs (even forbidden ones)
%L add all URL located in this text file (one URL per line)

The links options allow you to control what links are followed and what links are not as well as to provide long lists of links to investigate. Any setting other than the default for this option forces the engine to use less reliable and more complex parsing. 'Dirty' parsing means that links like 'xsgfd syaze="foo.gif"' will cause HTTrack to download foo.gif, even if HTTrack don't know what the "xsgfd syaze=" tag actually means! This option is powerful because some links might otherwise be missed, but it can cause errors in HTML or javascript.

This will direct the program to NOT search Javascript for unknown tag fields (e.g., it will find things like foo.location="bar.html"; but will not find things like bar="foo.gif";). While I have never had a reason to use this, some users may decide that they want to be more conservative in their searches. As a note, javascript imported files (.js) are not currently searched for URLs.

httrack http://www.shoesizes.com -O /tmp/shoesizes '%P0'

Now here is a classic bit of cleaverness that 'does the right thing' for some cases. In this instance, we are asking httrack to get images - like gif and jpeg files that are used by a web page in its display, even though we would not normally get them. For example, if we were only getting a portion of a web site (e.g., everything under the 'bob directory') we might want to get graphics from the rest of the web sote - or the rest of the web - that are used in those pages as well so that our mirror will look right.

httrack http://www.shoesizes.com -O /tmp/shoesizes -n

Here, we limit the collection to bob's area of the server - except that we get images and other such things that are used by bob in his area of the server.

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -n

This option 'tests' all links - even those forbidden
(by the robot exclusion protocol) - by using the 'HEAD' protocol to test
for the presence of a file.

httrack http://www.shoesizes.com/ -O /tmp/shoesizes -t

In this case, we use a file to list the URLs we wish
to mirror. This is particularly useful when we have a lot to do and
don't want to tirelessly type in URLs on command line after command line.
It's also useful - for example - if you update a set of mirrored sites
evey evening. You can set up a command like this to run automatically
from your cron file.

httrack -%L linkfile -O /tmp/shoesizes

This will update the mirror of your list of sites
whenever it is run.

httrack -%L linkfile -O /tmp/shoesizes -B --update

The link file is also useful for things like this
example where, after a binary image of a hard disk was analyzed (image)
URLs found on that disk were collected by httrack:

strings image | grep "http://" > list;
httrack -%L list -O /tmp/shoesizes

Mirror Build Options

Build options:
- ***NN name conversion type (0 *original structure, 1+: see below)***
- ***N user defined structure (-N "%h%p/%n%q.%t")***
- ***LN long names (L1 *long names / L0 8-3 conversion)***
- ***K keep original links (e.g. http://www.adr/link) (K0 *relative link)***
- ***x replace external html links by error pages***
- ***o *generate output html file in case of error (404..) (o0 don't generate)***
- ***X *purge old files after update (X0 keep delete)***
- ***%x do not include any password for external password protected websites (%x0 include) (--no-passwords)***
- ***%q *include query string for local files (information only) (%q0 don't include) (--include-query-string)***

The user can define naming conventions for building
the mirror of a site by using these options. For example, to retain the
original structure, the default is used. This only modifies the
structure to the extent that select characters (e.g., ~, :, <, >, \, and
@) are replaced by _ in all pathnames.

httrack http://www.shoesizes.com -O /tmp/shoesizes -N0

OR

httrack http://www.shoesizes.com -O /tmp/shoesizes

In either case, the mirror will build with the same
directory hierarchy and name structure as the original site. For cases
when you want to define your own structure, you use a string like this:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -N "%h%p/%n.%t"

In this case, %h, %p, $n, and %t stand for the href
element (e.g., http://www.shoesizes.com or ftp://ftp.shoesizes.com), %p
stands for the pathname (e.g., /bob/), %n stands for the name of the
file, and %t stands for type (file extension). The full list of these
options follows:

%n	Name of file without file type (ex: image)
%N	Name of file, including file type (ex: image.gif)
%t	File type (ex: gif)
%p	Path [without ending /] (ex: /someimages)
%h	Host name (ex: www.all.net)
%M	URL MD5 (128 bits, 32 ascii bytes)
%Q	query string MD5 (128 bits, 32 ascii bytes)
%q	small query string MD5 (16 bits, 4 ascii bytes)
%s?	Short name version (ex: %sN)

Other 'N' options include:

Details: Option N
- ***N0 Site-structure (default)***
- ***N1 HTML in web/, images/other files in web/images/***

N2 HTML in web/HTML, images/other in web/images
N3 HTML in web/, images/other in web/
N4 HTML in web/, images/other in web/xxx, where xxx is the file extension
(all gif will be placed onto web/gif, for example)
N5 Images/other in web/xxx and HTML in web/HTML
N99 All files in web/, with random names (gadget !)
N100 Site-structure, without www.domain.xxx/
N101 Identical to N1 exept that "web" is replaced by the site's name
N102 Identical to N2 exept that "web" is replaced by the site's name
N103 Identical to N3 exept that "web" is replaced by the site's name
N104 Identical to N4 exept that "web" is replaced by the site's name
N105 Identical to N5 exept that "web" is replaced by the site's name
N199 Identical to N99 exept that "web" is replaced by the site's name
N1001 Identical to N1 exept that there is no "web" directory
N1002 Identical to N2 exept that there is no "web" directory
N1003 Identical to N3 exept that there is no "web" directory (option set for g option)
N1004 Identical to N4 exept that there is no "web" directory
N1005 Identical to N5 exept that there is no "web" directory
N1099 Identical to N99 exept that there is no "web" directory

Long names are normally used (the ***-L0***
option) but if you are imaging to a DOS file system or want
accessibility from older versions of DOS and Windows, you can use the
-L1 option to generate these filename sizes.

httrack http://www.shoesizes.com -O /tmp/shoesizes -L1

With the 'K' option, you can keep the original links
in files. While this is less useful in being able to view a web site
froim the mirrored copy, it is vitally important if you want an accurate
copy of exactly what was on the web site in the first place. In a
forensic image, for example, you might want to use this option to
prevent the program from modifying the data as it is collected.

httrack http://www.shoesizes.com -O /tmp/shoesizes -K

In this case, instead of leaving external links (URLs
that point to sites not being mirrored) in the pages, these links are
replaced by pages that leave messages indicating that they could not be
found. This is useful for local mirrors not on the Internet or mirrors
that are on the Internet but that are not supposed to lead users to
external sites. A really good use for this is that 'bugging' devices
placed in web pages to track who is using them and from where will be
deactivated byt his process.

httrack http://www.shoesizes.com -O /tmp/shoesizes -x

This option prevents the generation of '404' error
files to replace files that were not found even though there were URLs
pointing to them. It is useful for saving space as well as eliminating
unnecessary files in operations where a working web site is not the
desired result.

httrack http://www.shoesizes.com -O /tmp/shoesizes -o0

This option prevents the authoatic purging of files
from the mirror site that were not found in the original web site after
an 'update' is done. If you want to retain old data and old names for
files that were renamed, this option should be used. If you want an
up-to-date reflection of the current web site, you should not use this option.

httrack http://www.shoesizes.com -O /tmp/shoesizes -X0

These options can be combined as desired to produce a
wide range of different arrangements, from collections of only graphical
files stored in a graphics area, to files identified by their MD5
checksums only, all stored in the same directory.

httrack http://www.shoesizes.com -O /tmp/shoesizes %x0 include

This will not include passwords for web sites. If you
mirror http://smith_john:foobar@www.privatefoo.com/smith/, and exclude
using filters some links, these links will be by default rewritten with
password data. For example, "bar.html" will be renamed into
http://smith_john:foobar@www.privatefoo.com/smith/bar.html This can be a
problem if you don't want to disclose the username/password! The %x
option tell the engine not to include username/password data in
rewritten URLs.

httrack http://www.shoesizes.com -O /tmp/shoesizes %q

This option is not very useful, because parameters are
useless, as pages are not dynamic anymore when mirrored. But some
javascript code may use the query string, and it can give useful
information. For example: catalog4FB8.html?page=computer-science is
clearer than catalog4FB8.html Therefore, this option is activated by
default.

Spider Options

These options provide for automation with regard to the remote server. For example, some sites require that cookies be accepted and sent back in order to allow access.

Spider options:

bN accept cookies in cookies.txt (0=do not accept,* 1=accept)
u check document type if unknown (cgi,asp..) (u0 don't check, * u1 check but /, u2 check always)
j *parse Java Classes (j0 don't parse)
sN follow robots.txt and meta robots tags (0=never,1=sometimes,* 2=always)
%h force HTTP/1.0 requests (reduce update features, only for old servers or proxies)
%B tolerant requests (accept bogus responses on some servers, but not standard!)
%s update hacks: various hacks to limit re-transfers when updating
%A assume that a type (cgi,asp..) is always linked with a mime type (-%A php3=text/html) (--assume)

By default, cookies are universally accepted and returned. This makes for more effective collection of data, but allows the site to be identified with its collection of data more easily. To disable cookies, use this option:

httrack http://www.shoesizes.com -O /tmp/shoesizes -b0

Some documents have known extension types (e.g., html), while others have unknown types (e.g., iuh87Zs) and others may have misleading types (e.g., an html file with a 'gif' file extension. These options provide for (0) not checking file types, (1) checking all file types except directories, and (2) checking all file types including directories. Choose from these options:

httrack http://www.shoesizes.com -O /tmp/shoesizes -u0
httrack http://www.shoesizes.com -O /tmp/shoesizes -u1
httrack http://www.shoesizes.com -O /tmp/shoesizes -u2

Meta tags or 'robots.txt' files on a web site are used to indicate what files should and should not be visited by automatic programs when collectiong data. The polite and prudent move for normal data collection (and the default) is to follow this indication:

httrack http://www.shoesizes.com -O /tmp/shoesizes -s2

This follows the robots protocol and meta-tags EXCEPT in cases where the filters disagree with the robots protocols or meta-tags.

httrack http://www.shoesizes.com -O /tmp/shoesizes -s1

In this next case, we ignore meta-tags and robots.txt files completely and just take whatever we can get from the site. The danger of this includes the fact that automated programs - like games or search engines may generate an unlimited number of nearly identical or identical outputs that will put us in an infinite loop collecting useless data under different names. The benefit is that we will get all the data there is to get.

httrack http://www.shoesizes.com -O /tmp/shoesizes -s0

This next option uses strict HTTP/1.0 protocol. This means the program will use HTTP/1.0 headers (as in RFC1945.TXT) and NOT extended 1.1 features described in RFC2616.TXT. For example, reget (complete a partially downloaded file) is a HTTP/1.1 feature. The Etag feature is also a HTTP/1.1 feature (Etag is a special identifier that allow to easily detect file changes).

httrack http://www.shoesizes.com -O /tmp/shoesizes -%h

Some servers give responses not strictly within the requirements of the official http protocol. These 'Bogus' responses can be accepted by using this option. For example, when requesting foo.gif (5132 bytes), the server can, optionally, add:

Content-length: 5132

This helps the client by allowing it to reserve a block of memory, instead of collecting each byte and re-reserving memory each time data is being received. But some servers are bogus, and send a wrong filesize. When HTtrack detects the end of file (connection broken), there are three cases:

1- The connection has been closed by the server, and we have received all data (we have received the number of bytes incicated by the server). This is fine because we have successfully received the

file.

2- The connection has been closed by the server, BUT the filesize received is different from the server's headers: the connection has been suddenly closed, due to network problems, so we reget the file

3- The connetion has been closed by the server, the filesize received is different from the server's headers, BUT the file is complete, because the server gave us a WRONG information! In this case, we use the bogus server option:

httrack http://www.shoesizes.com -O /tmp/shoesizes -%B

These options can be combined for the particular needs of the situaiton and are often adapted as a result of site-specific experiences.

httrack http://www.shoesizes.com -O /tmp/shoesizes -%s

This is a collection of "tricks" which are not really "RFC compliant" but which can save bandwidth by trying not to retransfer data in several cases.

httrack http://www.shoesizes.com -O /tmp/shoesizes -%A asp=text/html

The most important new feature for some people, maybe. This option tells the engine that if a link is en countered, with a specific type (.cgi, .asp, or .php3 for example), it MUST assume that this link has always the same MIME type, for example the "text/html" MIME type. This is VERY important to speed up many mirrors.

We have done tests on big HTML files (approx. 150 MB, 150,000,000 bytes!) with 100,000 links inside. Such files are being parsed in approx. 20 seconds on my own PC by the latest optimized releases of HTTra ck. But these tests have been done with links of known types, that is, html, gif, and so on.. If you have, say, 10,000 links of unknown type, such as ".asp", this will cause the engine to test ALL t hese files, and this will SLOOOOW down the parser. In this example, the parser will take hours, instead of 20 seconds! In this case, it would be great to tell HTTrack: ".asp pages are in fact HTML pages" This is possible, using: -%A asp=text/html

The -%A option can be replaced by the alias --assume asp=text/html which is MUCH more clear. You can use multiple definitions, separed by ",", or use multiple options. Therefore, these two lines are identical:

```
--assume asp=text/html --assume php3=text/html --assume cgi=image/gif
--assume asp=text/html,php3=text/html,cgi=image/gif
```

The MIME type is the standard well known "MIME" type. Here are the most important ones:

```
text/html       Html files, parsed by HTTrack
image/gif       GIF files
image/jpeg      Jpeg files
image/png       PNG files
```

There is also a collection of "non standard" MIME types. Example:

```
application/x-foo       Files with "foo" type
```

Therefore, you can give to all files terminated by ".mp3" the MIME type: application/x-mp3

This allow you to rename files on a mirror. If you KNOW that all "dat" files are in fact "zip" files ren amed into "dat", you can tell httrack:

```
--assume dat=application/x-zip
```

You can also "name" a file type, with its original MIME type, if this type is not known by HTTrack. This will avoid a test when the link will be reached:

```
--assume foo=application/foobar
```

In this case, HTTrack won't check the type, because it has learned that "foo" is a known type, or MIME type "application/foobar". Therefore, it will let untouched the "foo" type.

A last remark, you can use complex definitions like:

```
--assume asp=text/html,php3=text/html,cgi=image/gif,dat=application/x-zip,mpg=application/x-mp3,application/foobar
```

..and save it on your .httrackrc file:

```
set assume asp=text/html,php3=text/html,cgi=image/gif,dat=application/x-zip,mpg=application/x-mp3,application/foobar
```

Browser Options

Browsers commonly leave footprints in web servers - as
web servers leave footprints in the browser.

Browser ID:

- ***F user-agent field (-F "user-agent name")***
- ***%F footer string in Html code (-%F "Mirrored [from host %s [file %s [at %s]]]"***
- ***%l preffered language (-%l "fr, en, jp, *" (--language)***

The user-agent field is used by browsers to determine
what kind of browser you are using as well as other information - such
as your system type and operating system version. The 'User Agent'
field can be set to indicate whatever is desired to the server. In this
case, we are claiming to be a netscape browser (version 1.0) running a
non-exitent Solaris operating system version on a Sun Sparcstation.

httrack http://www.shoesizes.com -O /tmp/shoesizes -F "Mozilla 1.0, Sparc, Solaris 23.54.34"

On the other side, we may wish to mark each page
collected with footer information so that we can see from the page where
it was collected from, when, and under what name it was stored.

httrack http://www.shoesizes.com -O /tmp/shoesizes -%F "Mirrored [from host %s [file %s [at %s]]]"

This makes a modified copy of the file that may be
useful in future identification. While it is not 'pure' in some senses,
it may (or may not) be considered siilar to a camera that adds time and
date stamps from a legal perspective.

httrack http://www.shoesizes.com -O /tmp/shoesizes -%l "fr, en, jp, *"

"I prefer to have pages with french language, then
english, then japanese, then any other language"

Log, Cache, and Index Options

A lot of options are available for log files, indexing
of sites, and cached results:

Log, index, cache

- ***C create/use a cache for updates and retries (C0 no cache,C1 cache is prioritary,* C2 test update before)***
- ***k store all files in cache (not useful if files on disk)***
- ***%n do not re-download locally erased files***
- ***Q log quiet mode (no log)***
- ***q quiet mode (no questions)***
- ***z extra infos log***
- ***Z debug log***
- ***v verbose screen mode***
- ***%v display on screen filenames downloaded (in realtime) (--display)***
- ***f log file mode***
- ***f2 one single log file (--single-log)***
- ***I *make an index (I0 don't make)***
- ***%I make an searchable index for this mirror (* %I0 don't make) (--search-index)***

A cache memory area is used for updates and retries to
make the process far more efficient than it would otherwise be. You can
choose to (0) go without a cache, (1) do not check remotly if the file
has been updated or not, just load the cache content, or (2) see what
works best and use it (the default). Here is the no cache example.

httrack http://www.shoesizes.com -O /tmp/shoesizes -C0

The cache can be used to store all files - if desired
- but if files are being stored on disk anyway (the normal process for a
mirroring operation), this is not helpful.

httrack http://www.shoesizes.com -O /tmp/shoesizes -k

In some cases, a file from a mirror site is erased
locally. For example, if a file contains inappropriate content, it may
be erased from the mirror site but remain on the remote site. This
option allows you to leave deleted files permanently deleted when you
do a site update.

httrack http://www.shoesizes.com -O /tmp/shoesizes -update '%n'

If no log is desired, the following option should be added.

httrack http://www.shoesizes.com -O /tmp/shoesizes -Q

If no questions should be asked of the user (in a mode that would otherwise ask questions), the following option should be added.

httrack http://www.shoesizes.com -O /tmp/shoesizes -q

By adding these options, you get (-z) extra log information or (-Z) debugging information, and (-v) verbose screen output.

httrack http://www.shoesizes.com -O /tmp/shoesizes -z -Z -v

Multiple log files can be created, but by default, this option is used to put all logs into a single log file.

httrack http://www.shoesizes.com -O /tmp/shoesizes -f2

Finally, an index is normally made of the sites mirrored (a pointer to the first page found from each specified URL) in an index.html file in the project directory. This can be prevented through the use of this option:

httrack http://www.shoesizes.com -O /tmp/shoesizes -I0

httrack http://www.shoesizes.com -O /tmp/shoesizes %v

Animated information when using consol-based version, example:

17/95: localhost/manual/handler.html (6387 bytes) - OK

httrack http://www.shoesizes.com -O /tmp/shoesizes f2

Do not split error and information log (hts-log.txt and hts-err.txt) - use only one file (hts-log.txt)

httrack http://www.shoesizes.com -O /tmp/shoesizes -%I linux.localdomain

Still in testing, this option asks the engine to generate an index.txt, useable by third-party programs or scripts, to index all words contained in html files. The above example will produce index.txt:

```
..
abridged
        1 linux/manual/misc/API.html
        =1
        (0)
absence
        3 linux/manual/mod/core.html
        2 linux/manual/mod/mod_imap.html
        1 linux/manual/misc/nopgp.html
        1 linux/manual/mod/mod_proxy.html
        1 linux/manual/new_features_1_3.html
        =8
        (0)
absolute
        3 linux/manual/mod/mod_auth_digest.html
        1 linux/manual/mod/mod_so.html
        =4
        (0)
..
```

Expert User Options

For expert users, the following options provide further options.

Expert options:
pN priority mode: (* p3)
0 just scan, don't save anything (for checking links)
1 save only html files
2 save only non html files
****3 save all files***
7 get html files before, then treat other files
S stay on the same directory
D *can only go down into subdirs
U can only go to upper directories
B can both go up&down into the directory structure
a *stay on the same address
d stay on the same principal domain
l stay on the same location (.com, etc.)
e go everywhere on the web
%H debug HTTP headers in logfile

One interesting application allows the mirror utility to check for valid and invalid links on a site. This is commonly used in site tests to look for missing pages or other html errors. I often run such programs against my web sites to verify that nothing is missing.

httrack http://www.shoesizes.com -O /tmp/shoesizes -p0

To check for valid links outside of a site, the '-t' option can be used:

httrack http://www.shoesizes.com -O /tmp/shoesizes -t

These options can be combined, for example, to provide a service that checks sites for validity of links and reports back a list of missing files and statistics.

Other options allow the retention of select files - for example - (1) only html files, (2) only non-html files, (3) all files, and (7) get all html files first, then get other files. This last option provides a fast way to get the web pointers so that, for example, a time limited collection process will tend to get the most important content first.

In many cases, we only want the files froma given directory. In this case, we specify this option:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -S

This option allows the mirror to go only into subdirectories of the initial directory on the remote host. You might want to combine it with the ***-n*** option to get all non-html files linked from the pages you find.

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -D -n

If you only want to work your way up the directory structure from the specified URL (don't ask me why you might want to do this), the following command line is for you:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -U

If you want to go both up and down the directory structure (i.e., anywhere on on this site that the requested page leads you to), this option will be best:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -B

The default is to remain on the same IP address - or host name. This option specifes this explicitly:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -a

If you want to restrict yourself only to the same principal domain (e.g., include sites liks ftp.shoesizes.com), you would use this option.

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -d

To restrict yourself to the same major portion of the Internet (e.g., .com, .net, .edu, etc.) try this option:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -l

Finally, if you want to mirror the whole Internet - at least every place on the internet that is ever led to - either directly or indirectly - from the starting point, use this one... Please note that this will almost always run you out of resources unless you use other options - like limiting the depth of search.

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -e

Last but not least, you can include debugging informaiton on all headers from a collection process by using this option:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -'%H'

The options S, D, U, B, a, d, l, and e can be replaces with filter options approximately as follows:

S -www.foo.com/* +www.foo.com/bar/*[file]
D (default)
U +www.foo.com/bar/* -www.foo.com/*[name]/*
B +www.foo.com/bar/*
a (default)
d +*[name].foo.com/*
l +*[name].com/*
e +* (this is crazy unless a depth limit is used!)

Guru Options - DO NOT USE!!!

This is a new section, for all "not very well documented options". You can use them, in fact, do not believe what is written above!

#0 Filter test (-#0 '*.gif' 'www.bar.com/foo.gif')

To test the filter system. Example:

```
$ httrack -#0 'www.*.com/*foo*bar.gif' 'www.mysite.com/test/foo4bar.gif'
www.mysite.com/test/foo4bar.gif does match www.*.com/*foo*bar.gif
```

#f Always flush log files

Useful if you want the hts-log.txt file to be flushed regularly (not buffered)

#FN Maximum number of filters

Use if if you want to use more than the maximum default number of filters, that is, 500 filters: -#F2000 for 2,000 filters

#h Version info

Informations on the version number

#K Scan stdin (debug)

Not useful (debug only)

#L Maximum number of links (-#L1000000)

Use if if you want to use more than the maximum default number of links, that is, 100,000 links: -#L2000000 for 2,000,000 links

#p Display ugly progress information

Self-explanatory :) I will have to improve this one

#P Catch URL

"Catch URL" feature, allows to setup a temporary proxy to capture complex URLs, often linked with POST action (when using form based authentication)

#R Old FTP routines (debug)

Debug..

#T Generate transfer ops. log every minutes

Generate a log file with transfer statistics

#u Wait time

"On hold" option, in seconds

#Z Generate transfer rate statictics every minutes

Generate a log file with transfer statistics

#! Execute a shell command (-#! "echo hello")

Debug..

Command-line Specific Options

Command-line specific options:
*** V execute system command after each files ($0 is the filename: -V "rm \$0") (--userdef-cmd)***

This option is very nice for a wide array of actions
that might be based on file details. For example, a simple log of all
files collected could be generated by using:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes -V "/bin/echo \$0"

%U run the engine with another id when called as root (-%U smith) (--user)

Change the UID of the owner when running as r00t

  Details: User-defined option N
    %[param] param variable in query string

This new option is important: you can include query-string content when forming the destination filename!

Example: you are mirroring a huge website, with many pages named as:
www.foo.com/catalog.php3?page=engineering
www.foo.com/catalog.php3?page=biology
www.foo.com/catalog.php3?page=computing
..

Then you can use the -N option:

httrack www.foo.com -N "%h%p/%n%[page].%t"

If found, the "page" parameter will be included after
the filename, and the URLs above will be saved as:

/home/mywebsites/foo/www.foo.com/catalogengineering.php3
/home/mywebsites/foo/www.foo.com/catalogbiology.php3
/home/mywebsites/foo/www.foo.com/catalogcomputing.php3
...

Shortcuts

These options provide shortcust to combinations
of other options that are commonly used.

Shortcuts:
--mirror   *make a mirror of site(s) (default)
--get   get the files indicated, do not seek other URLs (-qg)
--list   add all URL located in this text file (-%L)
--mirrorlinks   mirror all links in 1st level pages (-Y)
--testlinks   test links in pages (-r1p0C0I0t)
--spider   spider site(s), to test links: reports Errors & Warnings (-p0C0I0t)
--testsite   identical to --spider
--skeleton   make a mirror, but gets only html files (-p1)
--update   update a mirror, without confirmation (-iC2)
--continue   continue a mirror, without confirmation (-iC1)
--catchurl   create a temporary proxy to capture an URL or a form post URL
--clean   erase cache & log files
--http10   force http/1.0 requests (-%h)

Mirror is the default behavior. It is detailed earlier.

get simply gets the files specified on the command line.

The list option is useful for including a list of sites to collect data from.

The mirrorlinks option is ideal for using the result of a previous search (like a list of pages found in a web search or somebody's URL collection) to guide the collection of data. With additional options (such as depth 1) it can be used to collect all of the pages linked to a given page without going further. Here is an example:

httrack http://www.shoesizes.com/bob/ -O /tmp/shoesizes --mirrorlinks -e -r1

Testing links in pages is useful for automating the verification that a link from a file is not pointing to a non-existent page.

The spider option does a site test automatically and returns errors for broken links.

The skeleton option makes a mirror of html files only.

The update option updates a site to match a remote mirror.

The continue option continues a previously terminated mirroring activity. This is useful for all sorts of mirror failures.

The catchurl option is a small application designed to catch difficult pages, like sites protected via formulas. You can see at http://httrack.free.fr/HelpHtml/addurl.html a Windows description of this application. The purpose is to create a temporary proxy, that will catch the user request to a page, and then store this request to continue the mirror. For example,

1. browse www.foo.com/bar/ until you have a page with a form
2. fill this form to enter the site BUT do not click "submit"
3. start the --catchurl application
4. change your browser proxy settings according to the --catchurl application
5. click on "submit" on your browser
6. HTTrack has now captured this click and has stored it
7. restore your proxy settings
8. (click back in your browser)

The clean option erases cache and log files.

The http10 option forces http/1.0 requests (the same as -%h).

Filters

Filters are normally placed at the end of the command line, but can be intermixed with other command line options if desired, except that if they are placed between (for example) the '-O' and the pathname, your results may be different than you might otherwise predict. There are two sorts of filters, filters that indicate what to include (+) and filters that indicate what to exclude (-).

Starting with the initially specified URLs, the default operation mode is to mirror starting from these URLs downward into the directory structure of the host (i.e. if one of your starting pagees was www.all.net/test/a.html, all links starting with www.all.net/test/ will be collected but links in www.all.net/anything-else will not be collected, because they are in a higher directory strcuture level. This prevents HTTrack from mirroring the whole site. If you may want to download files are in other parts of the site or pf particular types - or to not download files in a particular part of the site or of a particular type, you can use filters to specify more precisely what to collect and what not to collect.

The syntax for filters is similar to Unix regular expressions. A simple filter can be made by using characters from the URL with '*' as a wildcard for 0 or more characters - with the last filter rule having the highest precendence. An initial '+' indicates URLs to include and an initial '-' indicated URLs to not include. For example:

'-*' '+*jpg'

would only get files ending in the 'jpg' extension, while:

'-*jpg'

would not get any files ending in the jpg extension.
You can add more filter lines to restrict or expand the scope as desired. The last rule is checked first, and so on - so that the rules are in reverse priority order. Here's an example:

+*.gif -image*.gif	Will accept all gif files BUT image1.gif,imageblue.gif,imagery.gif and so on
-image*.gif +*.gif	Will accept all gif files, because the second pattern is prioritary (because it is defined AFTER the first one)

The full syntax for filters follows:

*****	any characters (the most commonly used)
***[file] or *[name]**	any filename or name, e.g. not /,? and ; characters
***[path]**	any path (and filename), e.g. not ? and ; characters
***[a,z,e,r,t,y]**	any letters among a,z,e,r,t,y
***[a-z]**	any letters
***[0-9,a,z,e,r,t,y]**	any characters among 0..9 and a,z,e,r,t,y
***[]**	no characters must be present after
***[< NN]**	size less than NN Kbytes
***[> PP]**	size more than PP Kbytes
***[< NN > PP]**	size less than NN Kbytes and more than PP Kbytes

Here are some examples of filters: (that can be generated automatically using the interface)

-www.all.net*	This will refuse/accept this web site (all links located in it will be rejected)
+*.com/*	This will accept all links that contains .com in them
-*cgi-bin*	This will refuse all links that contains cgi-bin in them
+*.com/*[path].zip	This will accept all zip files in .com addresses
-*someweb*/*.tar*	This will refuse all tar (or tar.gz etc.) files in hosts containing someweb
+*/*somepage*	This will accept all links containing somepage (but not in the address)
-*.html	This will refuse all html files from anywhere in the world.
+*.html*[]	Accept ***.html**, but the link must not have any supplemental characters at the end (e.g., links with parameters, like **www.all.net/index.html?page=10** will not match this filter)
-*.gif*[> 5] -*.zip +*.zip*[< 10]	refuse all gif files smaller than 5KB, exlude all zip files, EXCEPT zip files smaller than 10KB

User Authentication Protocols

Smoe servers require user ID and password information in order to gain access. In this example, the user ID smith with password foobar is accessing www.all.net/private/index.html

httrack smith:foobar@www.all.net/private/index.html

For more advanced forms of authentication, such as those involving forms and cookies of various sorts, an emerging capability is being provided through th URL capture features (--catchurl). This feature don't work all of the time.

.httrackrc

A file called '.httrackrc' can be placed in the current directory, or if not found there, in the home directory, to include command line options. These options are included whenever httrack is run. A sample .httrack follows:

```
set sockets 8
set retries 4
index on
set useragent "Mozilla [en] (foo)"
set proxy proxy:8080
```

But the syntax is not strict, you can use any of these:

```
set sockets 8
set sockets=8
sockets=8
sockets 8
```

.httrackrc is sought in the following sequence with the first occurence used:

- in the dirctory indicated by -O option (.httrackrc)
- in the current directory (.httrackrc)
- in the user's home directory (.httrackrc)
- in /etc/httrack.conf (named httrack.conf to be "standard")

An example .httrackrc looks like:

```
set sockets=8
set index on
retries=2
allow *.gif
```

```
deny ad.doubleclick.net/*
```

Each line is composed of an option name and a parameter. The "set" token can be used, but is not mandatory (it is ignored, in fact). The "=" is also optionnal, and is replaced by a space internally. The "on" and "off" are the same as "1" and "0" respectively. Therefore, the example .httrackrc above is equivalent to:

```
sockets=8
index=1
retries=2
allow=*.gif
deny=ad.doubleclick.net/*
```

Because the "=" seems to (wrongly) imply a variable assignment (the option can be defined more than once to define more than one filter) the following .httrackrc:

```
allow *.gif
allow *.jpg
```

looks better for a human than:

```
allow=*.gif
allow=*.jpg
```

Here's a example run with the example .httrackrc file:

```
$ httrack ghost
$ cat hts-cache/doit.log
-c8 -C1 -R2 +*.gif -ad.doubleclick.net/* ghost
```

The "-c8 -C1 -R2 +*.gif -ad.doubleclick.net/*" was added by the .httrackrc

---

Release Notes

Some things change between releases. Here are some recent changes in httrack that may affect some of these options:

Options S,D,U,B, and a,d,l,e are default behaviours of HTTrack. they were the only options in old versions (1.0). With the introduction of filters, their roles are now limited, because filters can override them.

Note for the -N option: "%h%p/%n%q.%t" will be now be used if possible. In normal cases, when a file does not have any parameters (www.foo.com/bar.gif) the %q option does not add anything, so there are no differences in file names. But when parameters are present (for example, www.foo.com/bar.cgi?FileCollection=133.gif), the additionnal query string (in this case, FileCollection=133.gif) will be "hashed" and added to the filename. For example:

```
'www.all.net/bar.cgi?FileCollection=133.gif'
```

will be named

```
'/tmp/mysite/bar4F2E.gif'
```

The additionnal 4 letters/digits are VERY useful in cases where there are a substantial number of identical files:

```
www.all.net/bar.cgi?FileCollection=133.gif
www.all.net/bar.cgi?FileCollection=rose.gif
www.all.net/bar.cgi?FileCollection=plant4.gif
www.all.net/bar.cgi?FileCollection=silver.gif
and so on...
```

In these cases, there is a small probability of a hash collision forlarge numbers of files.

Some More Examples

Here are some examples of special purpose httrack command lines that might be useful for your situation.

This is a 'forensic' dump of a web site - intended to collect all URLs reachable from the initial point and at that particular site. It is intended to make no changes whatsoever to the image. It also prints out an MD5 checksum of each file imaged so that the image can be verified later to detect and changes after imaging. It uses 5

retries to be more certain than normal of getting the files, never
abandons its efforts, keeps original links, does not generate error
files, ignores site restrictions for robots, logs as much as it can,
stays in the principal domain, places debugging headers in the log file,

httrack "www.website.com/" -O "/tmp/www.website.com" -R5H0Ko0s0zZd %H -V "md5 \$0" "+*.website.com/*"

Here's an example of a site where I pulled a set of
data related to some subject. In this case, I only wanted the
relevant subdirectory, all external links were to remain the same, a
verbose listing of URLs was to be printed, and I wanted files near (n)
and below (D) the original directory. Five retries just makes sure I
don't miss anything.

httrack "http://www.somesite.com/~library/thing/thingmain.htm" -O /tmp/thing -R5s0zZvDn

This listing is, of course, rather verbose. To reduce the noise,
you might want to do something more like this:

httrack "http://www.somesite.com/~library/thing/thingmain.htm" -O /tmp/thing -R5s0zvDn

A still quieter version - without any debugging
information but with a list of files loaded looks like this:

httrack "http://www.somesite.com/~library/thing/thingmain.htm" -O /tmp/thing -R5s0vDn

For the strong silent type, this might be still better:

httrack "http://www.somesite.com/~library/thing/thingmain.htm" -O /tmp/thing -R5s0qDn

---

General questions:

Q: The install is not working on NT without administrator rights!

A: That's right. You can, however, install WinHTTrack
on your own machine, and then copy your **WinHTTrack** folder from
your **Program Files** folder to another machine, in a temporary
directory (e.g. **C:\temp**)

Q: Where can I find French/other languages documentation?

A: Windows interface is available on several
languages, but not yet the documentation!

Q: Is HTTrack working on NT/2000?

A: Yes, it should

Q: What's the difference between HTTrack and WinHTTrack?

A: WinHTTrack is the Windows release of HTTrack (with
a graphic shell)

Q: Is HTTrack Mac compatible?

A: No, because of a lack of time. But sources are
available

Q: Can HTTrack be compiled on all Un*x?

A: It should. The **Makefile** may be modified in
some cases, however

**Q: I use HTTrack for professional purpose. What
about restrictions/license fee?**

A: There is no restrictions using HTTrack for
professional purpose, except if you want to sell a product including
HTTrack components (parts of the source, or any other component). See
the **license.txt** file for more informations

Q: Is a DLL/library version available?

A: Not yet. But, again, sources are available (see
license.txt for distribution infos)

**Q: Is there a X11/KDE shell available for Linux and
Un*x?**

A: No. Unfortunately, we do not have enough time for
that - if you want to help us, please write one!

Troubleshooting:

Q: Only the first page is caught. What's wrong?

A: First, check the **hts-err.txt** error log file - this can give you precious informations.

The problem can be a website that redirects you to another site (for example, **www.all.net** to **public.www.all.net**) : in this case, use filters to accept this site

This can be, also, a problem in the HTTrack options (link depth too low, for example)

Q: With WinHTTrack, sometimes the minimize in system tray causes a crash!

A: This bug sometimes appears in the shell on some systems. If you encounter this problem, avoid minimizing the window!

Q: Files are created with strange names, like '-1.html'!

A: Check the build options (you may have selected user-defined structure with wrong parameters!)

Q: When capturing real audio links (.ra), I only get a shortcut!

A: Yes. The audio/video realtime streaming capture is not yet supported

Q: Using user:password@address is not working!

A: Again, first check the **hts-err.txt** error log file - this can give you precious informations

The site may have a different authentication scheme (form based authentication, for example)

Q: When I use HTTrack, nothing is mirrored (no files) What's happening?

A: First, be sure that the URL typed is correct. Then, check if you need to use a proxy server (see proxy options in WinHTTrack or the **-P proxy:port** option in the command line program). The site you want to mirror may only accept certain browsers. You can change your "browser identity" with the Browser ID option in the OPTION box. Finally, you can have a look at the hts-err.txt (and hts-log.txt) file to see what happened.

Q: There are missing files! What's happening?

A: You may want to capture files that are in a different folder, or in another web site. In this case, HTTrack does not capture them automatically, you have to ask it to do. For that, use the filters.

Example: You are downloading **http://www.all.net/foo/** and can not get .jpg images located in **http://www.all.net/bar/** (for example, http://www.all.net/bar/blue.jpg)

Then, add the filter rule **+www.all.net/bar/*.jpg** to accept all .jpg files from this location

You can, also, accept all files from the /bar folder with **+www.all.net/bar/***, or only html files with **+www.all.net/bar/*.html** and so on..

Q: I'm downloading too many files! What can I do?

A: This is often the case when you use too large filters, for example **+*.html**, which asks the engine to catch all .html pages (even ones on other sites!). In this case, try to use more specific filters, like **+www.all.net/specificfolder/*.html**

If you still have too many files, use filters to avoid somes files. For example, if you have too many files from www.all.net/big/, use **-www.all.net/big/*** to avoid all files from this folder.

Q: File types are sometimes changed! Why?

A: By default, HTTrack tries to know the type of remote files. This is useful when links like **http://www.all.net/foo.cgi?id=1** can be either HTML pages, images or anything else. Locally, foo.cgi will not be recognized as an html page, or as an image, by your browser. HTTrack has to rename the file as foo.html or foo.gif so that it can be viewed.

Sometimes, however, some data files are seen by the remote server as html files, or images : in this case HTTrack is being fooled.. and rename the file. You can avoid this by disabling the type checking in the option panel.

Q: I can not access to several pages (access forbidden, or redirect to another location), but I can with my browser, what's going on?

A: You may need cookies! Cookies are specific datas (for example, your username or password) that are sent to your browser once you have logged in certain sites so that you only have to log-in once. For example, after having entered your username in a website, you can view pages and articles, and the next time you will go to this site, you will not have to re-enter your username/password.

To "merge" your personnal cookies to an HTTrack project, just copy the cookies.txt file from your Netscape folder (or the cookies located into the Temporary Internet Files folder for IE) into your project folder (or even the HTTrack folder)

Q: Some pages can't be seen, or are displayed with errors!

A: Some pages may include javascript or java files that are not recognized. For example, generated filenames. There may be transfer problems, too (broken pipe, etc.). But most mirrors do work. We still are working to improve the mirror quality of HTTrack.

Q: Some Java applets do not work properly!

A: Java applets may not work in some cases, for example if HTTrack failed to detect all included classes or files called within the class file. Sometimes, Java applets need to be online, because remote files are directly caught. Finally, the site structure can be incompatible with the class (always try to keep the original site structure when you want to get Java classes)

If there is no way to make some classes work properly, you can exclude them with the filters. They will be available, but only online.

Q: HTTrack is being idle for a long time without transfering. What's happening?

A: Maybe you try to reach some very slow sites. Try a lower TimeOut value (see options, or **-Txx** option in the command line program). Note that you will abandon the entire site (except if the option is unchecked) if a timeout happen You can, with the Shell version, skip some slow files, too.

Q: I want to update a site, but it's taking too much time! What's happening?

A: First, HTTrack always tries to minimize the download flow by interrogating the server about the file changes. But, because HTTrack has to rescan all files from the begining to rebuild the local site structure, it can takes some time. Besides, some servers are not very smart and always consider that they get newer files, forcing HTTrack to reload them, even if no changes have been made!

Q: I am behind a firewall. What can I do?

A: You need to use a proxy, too. Ask your administrator to know the proxy server's name/port. Then, use the proxy field in HTTrack or use the **-P proxy:port** option in the command line program.

Q: HTTrack has crashed during a mirror, what's happening?

A: We are trying to avoid bugs and problems so that the program can be as reliable as possible. But we can not be infallible. If you occurs a bug, please check if you have the latest release of HTTrack, and send us an email with a detailed description of your problem (OS type, addresses concerned, crash description, and everything you deem to be necessary). This may help the other users too.

Q: I want to update a mirrored project, but HTTrack is retransfering all pages. What's going on?

A: First, HTTrack always rescan all local pages to reconstitute the website structure, and it can take some time. Then, it asks the server if the files that are stored locally are up-to-date. On most sites, pages are not updated frequently, and the update process is fast. But some sites have dynamically-generated pages that are considered as "newer" than the local ones.. even if there are identical! Unfortunately, there is no possibility to avoid this problem, which is strongly linked with the server abilities.

Questions concerning a mirror:

Q: I want to mirror a Web site, but there are some files outside the domain, too. How to retrieve them?

A: If you just want to retrieve files that can be reached through links, just activate the 'get file near links' option. But if you want to retrieve html pages too, you can both use wildcards or explicit addresses ; e.g. add **www.all.net/*** to accept all files and pages from www.all.net.

Q: I have forgotten some URLs of files during a long mirror.. Should I redo all?

A: No, if you have kept the 'cache' files (in hts-cache), cached files will not be retransfered.

Q: I just want to retrieve all ZIP files or other files in a web site/in a page. How do I do it?

A: You can use different methods. You can use the 'get files near a link' option if files are in a foreign domain. You can use, too, a filter adress: adding **+*.zip** in the URL list (or in the filter list) will accept all ZIP files, even if these files are outside the address.

Example : **httrack www.all.net/someaddress.html +*.zip** will allow you to retrieve all zip files that are linked on the site.

Q: There are ZIP files in a page, but I don't want to transfer them. How do I do it?

A: Just filter them: add **-*.zip** in the filter list.

Q: I don't want to load gif files.. but what may happen if I watch the page?

A: If you have filtered gif files (**-*.gif**), links to gif files will be rebuild so that your browser can find them on the server.

Q: I get all types of files on a web site, but I didn't select them on filters!

A: By default, HTTrack retrieves all types of files on authorized links. To avoid that, define filters like

-* +<website>/*.html +<website>/*.htm +<website>/ +*.<type wanted>

Example: **httrack www.all.net/index.html -* +www.all.net/*.htm* +www.all.net/*.gif +www.all.net/*.jpg**

Q: When I use filters, I get too many files!

A: You are using too large a filter, for example ***.html** will get ALL html files identified. If you want to get all files on an address, use **www.<address>/*.html**. There are lots of possibilities using filters.

Example:**httrack www.all.net +*.www.all.net/*.htm***

Q: When I use filters, I can't access another domain, but I have filtered it!

A: You may have done a mistake declaring filters, for example **+www.all.net/* -*all*** will not work, because -*all* has an upper priority (because it has been declared after +www.all.net)

Q: Must I add a '+' or '-' in the filter list when I want to use filters?

A: YES. '+' is for accepting links and '-' to avoid them. If you forget it, HTTrack will consider that you want to accept a filter if there is a wild card in the syntax - e.g. +<filter> if identical to <filter> if <filter> contains a wild card (*) (else it will be considered as a normal link to mirror)

Q: I want to find file(s) in a web-site. How do I do it?

A: You can use the filters: forbid all files (add a **-*** in the filter list) and accept only html files and the file(s) you want to retrieve (BUT do not forget to add **+<website>*.html** in the filter list, or pages will not be scanned! Add the name of files you want with a ***/** before ; i.e. if you want to retrieve file.zip, add ***/file.zip**)

Example:**httrack www.all.net +www.all.net/*.htm* +thefileiwant.zip**

Q: I want to download ftp files/ftp site. How to do?

A: First, HTTrack is not the best tool to download many ftp files. Its ftp engine is basic (even if reget are possible) and if your purpose is to download a complete site, use a specific client.

You can download ftp files just by typing the URL, such as **ftp://ftp.www.all.net/pub/files/file010.zip** and list ftp directories like **ftp://ftp.www.all.net/pub/files/** .

Note: For the filters, use something like **+ftp://ftp.www.all.net/***

Q: How can I retrieve .asp or .cgi sources instead of .html result?

A: You can't! For security reasons, web servers do not allow that.

Q: How can I remove these annoying <!--

Mirrored from... --> from html files?

A: Use the footer option (-&F, or see the WinHTTrack options)

Q: Do I have to select between ascii/binary transfer mode?

A: No, http files are always transfered as binary files. Ftp files, too (even if ascii mode could be selected)

Q: Can HTTrack perform form-based authentication?

A: Yes. See the URL capture abilities (--catchurl for command-line release, or in the WinHTTrack interface)

Q: Can I redirect downloads to tar/zip archive?

A: Yes. See the shell system command option (-V option for command-line release)

Q: Can I use username/password authentication on a site?

A: Yes. Use user:password@your_url (example: **http://foo:bar@www.all.net/private/mybox.html**)

Q: Can I use username/password authentication for a proxy?

A: Yes. Use user:password@your_proxy_name as your proxy name (example: **smith:foo@proxy.mycorp.com**)

Q: Can HTTrack generates HP-UX or ISO9660 compatible files?

A: Yes. See the build options (-N, or see the WinHTTrack options)

Q: If there any SOCKS support?

A: Not yet!

Q: What's this hts-cache directory? Can I remove it?

A: NO if you want to update the site, because this directory is used by HTTrack for this purpose. If you remove it, options and URLs will not be available for updating the site

Q: Can I start a mirror from my bookmarks?

A: Yes. Drag&Drop your bookmark.html file to the WinHTTrack window (or use file://filename for command-line release) and select bookmark mirroring (mirror all links in pages, -Y) or bookmark testing (--testlinks)

Q: I am getting a "pipe broken" error and the mirror stops, what should I do?

A: Chances are this is a result of downloading too many pages at a time. Remote servers may not allow or be able to handle too many sessions, or your system may be unable to provide the necessary resources. Try redusing this number - for example using the -c2 options for only 2 simultaneous sesions.