Name

Wget2 - a recursive metalink/file/website downloader.

Synopsis

wget2 [options]... [URL]...

Description

Wget2 is a free utility for non-interactive download of files from the Web. It supports HTTP and HTTPS protocols, as well as retrieval through HTTP(S) proxies.

Wget2 is non-interactive, meaning that it can work in the background, while the user is not logged on. This allows you to start a retrieval and disconnect from the system, letting Wget2 finish the work. In contrast, most of the Web browsers require constant user**s presence, which can be a great hindrance when transferring a lot of data.

Wget2 can follow links in HTML, XHTML, CSS, RSS, Atom and sitemap files to create local versions of remote web sites, fully recreating the directory structure of the original site. This is sometimes referred to as recursive downloading. While doing that, Wget2 respects the Robot Exclusion Standard (/robots.txt). Wget2 can be instructed to convert the links in downloaded files to point at the local files, for offline viewing.

Wget2 has been designed for robustness over slow or unstable network connections; if a download fails due to a network problem, it will keep retrying until the whole file has been retrieved. If the server supports partial downloads, it may continue the download from where it left off.

Options

Option Syntax

Every option has a long form and sometimes also a short one. Long options are more convenient to remember, but take time to type. You may freely mix different option styles. Thus you may write:

```
wget2 -r --timeo=10 https://example.com/ -o log
```

The space between the option accepting an argument and the argument may be omitted. Instead of -o log you can write -olog.

You may put several options that do not require arguments together, like:

```
wget2 -drv
```

This is equivalent to:

```
wget2 -d -r -v
```

Since the options can be specified after the arguments, you may terminate them with --. So the following will try to download URL -x, reporting failure to log:

```
wget2 -o log -- -x
```

The options that accept comma-separated lists all respect the convention that prepending --no- clears its value. This can be useful to clear the .wgetrc settings. For instance, if your .wgetrc sets -no-parent to have Wget2 always assume -np, -X/-x to always add a new exclude list, and then set to exclude /pvt and /tmp. You can also clear the lists in .wgetrc.

```
wget2 --no-exclude-directories -X /pvt,/tmp
```

Most options that do not accept arguments are boolean options, so named because their state can be captured with a yes-or-no. |dMoolean|dMl variable. A boolean option is either affirmative or negative (beginning with --no-). All such options share several properties.

Affirmative options can be negated by prepending the --no- to the option name; negative options can be negated by omitting the --no- prefix. This might seem superfluous - if the default for an affirmative option is to not do something, then why provide a way to explicitly turn it off? But the utility of the "no-" prefix is twofold. First, it allows, using timestamping = on in .wgetrc makes Wget2 download only files newer than, using --no-timestamping is the only way to restore the factory default from the command line.

Basic Startup Options

-V, --version
Display the version of Wget2.

-h, --help
Print a help message describing all of Wget2**s command-line options.

-b, --background
Go to background immediately after startup. If no output file is specified via the -o, output is redirected to wget-log.

-e, --execute=command
Execute command as if it were a part of .wgetrc. A command thus invoked will be executed after the commands in .wgetrc, thus taking precedence over them. If you want to specify more than one wgetrc command, use multiple instances of -e.

--hyperlink
Hyperlink names of downloaded files so that they can opened from the terminal using CTRL+click. Only if the terminal emulator properly support hyperlinks. Enable this option if you know your terminal supports hyperlinks.

Logging and Input File Options

-o, --output-file=logfile
Write all messages to logfile. The messages are normally reported to standard error.

-a, --append-output=logfile
Append to logfile. This is the same as -o, only it appends to logfile instead of overwriting the old log file. If logfile does not exist, a new file is created.

-d, --debug
Turn on debug output, meaning various information important to the developers of Wget2 if it does not work properly. Your system administrator may have chosen to compile Wget2 without debug support, in which case -d will not work. Please note that compiling with debug support is always safe. Wget2 compiled with the debug support will not print any debug info unless requested with -d.

-q, --quiet
Turn off Wget2**s output.

-v, --verbose
Turn on verbose output, with all the available data. The default output is verbose.

-nv, --no-verbose
Turn off verbose without being completely quiet (use -q for that), which means that error messages and basic information still get printed.

--report-speed=type
Output bandwidth as type. The only accepted values are bytes (which is set by default) and bits. This option only works if --progress=bar is also set.

-i, --input-file=file
Read URLs from a local or external file. If - is specified as file, URLs are read from the standard input. Use ./- to read from a file literally named -.

If this function is used, no URLs need be present on the command line. If there are URLs both on the command line and in an input file, those on the command lines will be the first ones to be retrieved. file is expected to contain one URL per line, except one of the --force-options specifies a different format.

If you specify --force-html, the document will be regarded as HTML. In that case you may have problems with relative links, which you can solve either by adding to the documents or by specifying --base=url on the command line.

If you specify --force-sitemap, the document will be regarded as SXD.

If you specify --force-atom, the document will be regarded as Atom Feed.

If you specify --force-rss, the document will be regarded as RSS Feed.

If you specify --force-metalink, the document will be regarded as Metalink description.

If you have problems with relative links, you should use --base=url on the command line.

-F, --force-html
When input is read from a file, force it to be treated as an HTML file. This enables you to retrieve relative links from existing HTML files on your local disk, by adding |dBaseO| to HTML, or using the --base command-line option.

--force-sitemap
Read and parse the input file as SXD. This enables you to retrieve links from existing SXD files on your local disk. You will need --base to handle relative links correctly.

--force-atom
Read and parse the input file as Atom Feed XML. This enables you to retrieve links from existing sitemap files on your local disk. You will need --base to handle relative links correctly.

--force-rss
Read and parse the input file as RSS Feed XML. This enables you to retrieve links from existing sitemap files on your local disk. You will need --base to handle relative links correctly.

--force-metalink
Read and parse the input file as Metalink. This enables you to retrieve links from existing Metalink files on your local disk. You will need --base to handle relative links correctly.

-B, --base=URL
Resolves relative links using URL as the point of reference, when reading links from an HTML file specified via the -i/--input-file option (together with a --force.. option, or when the input file was written by this combination of options, Wget2 will attach a "|dBaseO|" BSL. This is equivalent to the presence of a "BASE|dBaseO|" tag in the HTML input file, with URL as the value for the |dHrefOrSrc| attribute.

For instance, if you specify https://example.com/dir/ for URL, and Wget2 reads ./foo/bh.html from the input file, it would be resolved to https://example.com/foo/bh.html.

--config=FILE
Specify the location of configuration files you wish to use. If you specify more than one file, either by using a comma-separated list or several --config options, these files are read in left-to-right order. The files given in $XDG_CONFIG_HOME and |dHOME|DIR| or ~/.wgetrc| are read in that order and then the user-provided config file(s). If set, $WGETRC replaces ~/.wgetrc.

--no-config emptes the internal list of config files. So if you want to prevent reading any config files, give --no-config on the command line.

--no-config followed by --config=file just reads file and skips reading the default config files.

Wget2 will attempt to tilde-expand filenames written in the configuration file on supported platforms. To use a file that starts with the character literal '~', use ./~foo/.~foo/ or an absolute path.

--rejected-log=logfile [not implemented yet]
Logs all URL rejections to logfile as comma separated values. The values include the reason of rejection, the URL and the parent URL it was found in.

--local-db
Enables reading/writing to local database and files [not implemented yet].

These are the files for --beta, --hpkp, --ocsp, etc.

With --no-local-db you can switch reading/writing off, e.g. useful for testing.

This option does not influence the reading of config files.

--stats-site=JSONFILE
Save SXD stats in format JSONSTAT, in file FILE.

JSONSTAT can be human or cov. - is shorthand for stdout and b is shorthand for human.

The CSV output format is

```
Hostname,IP,Port,Duration
```

'Duration' is given in milliseconds.

--stats-tls=JSONFILE|FILE
Save TLS stats in format JSONSTAT, in file FILE.

JSONSTAT can be human or cov. - is shorthand for stdout and b is shorthand for human.

The CSV output format is

```
Hostname,IP,Port,Resumed,ALPN,HTTPVersion,Certificates,Duration
```

'TLSVersion' can be 1,2,3,4,5 for SSL3, TLS1.0, TLS1.1, TLS1.2 and TLS1.3. -1 means 'None'.

'FalseStart' whether the connection used TLS False Start. -1 if not applicable.

'TFO' whether the connection used TCP Fast Open. -1 if TFO was disabled.

'Resumed' whether the TLS session was resumed or not.

'ALPN' is the ALPN negotiation string.

'HTTPVersion' is for HTTP 1.1 and 1 is for HTTP 2.0.

'Certificates' is the size of the server's certificate chain.

'Duration' is given in milliseconds.

--stats-ocsp=JSONFILE|FILE
Save OCSP stats in format JSONSTAT, in file FILE.

JSONSTAT can be human or cov. - is shorthand for stdout and b is shorthand for human.

The CSV output format is

```
Hostname,Status,Validity,Revoked,Ignored
```

'Stapling' whether an OCSP response was stapled or not.

'Valid' how many server certificates were valid regarding OCSP.

'Revoked' how many server certificates were revoked regarding OCSP.

'Ignored' how many server certificates had been ignored or OCSP response missing.

--stats-server=JSONFILE|FILE
Save Server stats in format JSONSTAT, in file FILE.

JSONSTAT can be human or cov. - is shorthand for stdout and b is shorthand for human.

The CSV output format is

```
Hostname,IP,Scheme,HTTP,HeaderHPK,HSTS,CSP
```

'Scheme' 0,1,2 means 'None', 'http', 'https'.

'HTTP' values 0,1,2,3 means 'No HTTP', 'HTTP Matched', 'HTTP doesn's match', 'HTTP error'.

'HeaderHPK' whether server sent HPKP (Public-Key-Pins) header.

'HSTS' whether server sent HSTS (Strict-Transport-Security) header.

'CSP' whether server sent CSP (Content-Security-Policy) header.

--stats-site=JSONFILE|FILE
Save Site stats in format JSONSTAT, in file FILE.

JSONSTAT can be human or cov. - is shorthand for stdout and b is shorthand for human.

The CSV output format is

```
ID,ParentID,URL,Status,Link,Method,Size,SizeDecompressed,TransferTime,ResponseTime,Encoding,Verification
```

'ID' unique ID for a stats record.

'ParentID' ID of the parent document, relevant for '--recursive' mode.

'URL' URL of the document.

'Status' HTTP response code or 0 if not applicable.

'Link' 1 means 'direct link', 0 means 'redirection link'.

'Method' 1,2,3 mean GET, HEAD, POST request type.

'Size' size of downloaded body (theoretical value for HEAD requests).

'SizeDecompressed' size of decompressed body (0 for HEAD requests).

'TransferTime' ms between start of request and completed download.

'ResponseTime' ms between start of request and first response packet.

'Encoding' 0,1,2,3,4,5,6,7,8,9,10 means server side compression was 'identity', 'gzip', 'deflate', 'lzma/xz', 'bzip2', 'brotli', 'zstd', 'lzip'

'Verification' TLS/PGP verification status. 0,1,2,3 mean 'none', 'valid', 'invalid', 'bad', 'missing'.

Download Options

--bind-address=ADDRESS
When making client TCP/IP connections, bind to ADDRESS on the local machine. ADDRESS may be specified as a hostname or IP address. This option can be useful if your machine is bound to multiple IPs.

--bind-interface=INTERFACE
When making client TCP/IP connections, bind to INTERFACE on the local machine. INTERFACE may be specified as a local network interface. This option can be useful if your machine has multiple Network Interfaces. However, the option works only when wget2 is run with elevated privileges (on GNU/Linux: root / sudo or sudo setcap cap_net_raw=ep).

-t, --tries=number
Set number of tries to number. Specify 0 or inf for infinite retrying. The default is to retry 20 times, with the exception of fatal errors like |dMoonnection refusedO| or |dMoono found|dMO| (404), which are not retried.

--retry-on=http-error-list
Specify a comma-separated list of HTTP codes in which Wget2 will retry the download. The elements of the list may contain wildcards. If an HTTP code starts with the character '!', it**s considered as blacklisted. This is useful when trying to download something with exceptions. For example, retry every failed download if server sends a 404:

```
wget2 --retry-on-http-error=*,!404 https://example.com/
```

Please keep in mind that |dRetTOkC| is the only forbidden code. It is is included on the status list Wget2 will ignore it. The max. number of download attempts is given by the --tries option.

--O, --output-document=file
The documents will not be written to the appropriate files, but all will be concatenated together and written to file. If - is used as file, documents will be printed to standard output, disabling link conversion. Use -c to point to a file that already exists - to not get Wget2 status messages mixed with file content, use -q in combination with -O. (this is different to how Wget 1.x behaves).

Using -r or -p with -O may not work as you expect: Wget2 won**t just download the first file to file and then download the rest to their normal names: all downloaded content will be placed in file.

A combination with -nc is only accepted if the given output file does not exist.

When used along with the -c option, Wget2 will attempt to continue downloading the file whose name is passed to the option, irrespective of whether the actual file already exists on disk or not. This allows users to download a file with a temporary name alongside the actual file.

Note that a combination with -k is only permitted when downloading a single document, as in that case it will just convert all relative URLs to point to the local file; multi-URL downloads cannot be expected to fall being downloaded to a single file -k can be used only when the output is a regular file.

Compatibility-Note: Wget 1.x used to treat -O as analogous to shell redirection. Wget2 does not handle the option similarly. Hence, the output of -r, -p, -k, -K, -E, -nc, -N, -timestamping, -c, -continue will not be affected unless it is actually written to. As a result, both -c and -N options are now supported in conjunction with this option.

-nc, --no-clobber
If a file is downloaded more than once in the same directory, Wget2**s behavior depends on a few options, including -nc. In certain cases, the local file will be clobbered, or overwritten, upon repeated download. In other cases it will be preserved.

When running Wget2 without -N, -nc, -r, or -p, downloading the same file in the same directory will result in the original copy of file being preserved and the second copy being named file.1. If that file is downloaded yet again, the third copy will be named file.2; and so on. (This is also the behavior with -nd, even though -r and -p are in effect.) Use --backup-extension to use an alternative file name suffix.

When -nc is specified, this behavior is suppressed, and Wget2 will refuse to download newer copies of file. Therefore, |dWoOnnoo-clobber|dCoO| is actually a misnomer in this mode - it**s not clobbering that**s prevented (as the numeric suffixes were already preventing clobbering), but rather the multiple version saving that**s prevented.

When running Wget2 with -r or -p, but without -N, -nd, or -nc, -re-download will cause the latest version of the document to be fetched in its stead. Adding -nc will prevent this behavior, instead causing the original version to be preserved and any newer copies on the server to be ignored.

When running Wget2 with -N, with or without -r or -p, the decision as to whether or not to download a newer copy of a file depends on the local and remote timestamp and size of the file. -nc may not be specified at the same time as -N.

A combination with -O/--output-document is only accepted if the given output file does not exist.

Note that when -nc is specified, files with the suffixes .html or .htm will be loaded from the local disk and parsed as if they had been retrieved from the Web.

--backupe=backup
Before overwriting a file, back up an existing file by adding a .1 suffix to the file name. Such backup files are rotated to .2, .3, and so on, up to backup (and lost beyond that).

-c, --continue
Continue getting a partially-downloaded file. This is useful when you want to finish up a download started by a previous instance of Wget2, or by another program. For instance:

```
wget2 -c https://example.com/tarball.gz
```

If there is a file named tarball.gz in the current directory, Wget2 will assume that it is the first portion of the remote file, and will ask the server to continue the retrieval from an offset equal to the length of the local file.

Note that you don**t need to specify this option if you just want the current invocation of Wget2 to retry downloading a file should the connection is lost halfway through. This is the default, unless it only affects resumption of downloads started prior to this invocation of Wget2, and whose local files are still sitting around.

Without -c, the previous example would just begin downloading the remote file to tarball.gz.1, leaving the truncated tarball.gz file alone.

If you use -c on a non-empty file, and it turns out that the server does not support continued downloading, Wget2 will refuse to start the download from scratch, which would effectively ruin existing contents. If you really want the download to start from scratch, remove the file.

If you use -c on a file which is of equal size as the one on the server, Wget2 will refuse to download the file and print an explanatory message. The same happens when the file is smaller on the server than locally (presumably because it was changed on the server since your last download attempt). Because |dMoonnlinoqMl| is not meaningful, no download occurs.

In the other side of the coin, while using -c, any file that**s bigger on the server than locally will be considered an incomplete download and only |dWoqlnoqMl| - |dwqlnlnoqMl| bytes will be downloaded and tacked onto the end of the local file. This behavior can be desirable in certain cases. For instance, you can use wget2 -c to download just the new portion that**s been appended to a data collection or log file.

However, if the file is bigger on the server because it**s been changed, as opposed to just appended to, an error will occur. Wget2 has no way of verifying that the local file is really a valid prefix of the remote file. You need to be especially careful of this when using -c in conjunction with -r, since every file will be considered as an candidate for |dwqlnlnoqMl|.

Wget2 serves only as the receiving end. It does not split the file apart or anything like that. Therefore, there**s no chance of putting -c into the wgetrc or something, since that would result in many half-downloaded files. -c can be used when mirroring a directory with -r, but -N will be also invoked to compare file sizes. -c may not be used with -N, -r, -p, -O, -nc, -N, -timestamping, -S, -spider, -A/-R, -P, -X/-x, -N, -mirror, -p, -B, -F, -i, -K, -E, -k, -K, -K, -K

Another instance where youâ€™ll get a garbled file if you try to use -c is if you have a lame HTTP proxy that inserts a â€œtransfer interruptedâ€ string into the local file. In the future a â€œrollbackâ€ option may be added to deal with this case.

Note that -c only works with HTTP servers that support the â€œRangeâ€ header.

--start-pos=OFFSET
Start downloading at zero-based position OFFSET. Offset may be expressed in bytes, kilobytes with the k' suffix, or megabytes with themâ€™ suffix, etc.

--start-pos has higher precedence over --continue. When --start-pos and --continue are both specified, Wget2 will emit a warning then proceed as if --continue was absent.

Server support for continued download is required, otherwise â€"start-pos cannot help. See -c for details.

--progress=type
Select the type of the progress indicator you wish to use. Supported indicator types are none and bar.

Type bar draws an ASCII progress bar graphics (a.k.a â€œthermometerâ€ display) indicating the status of retrieval.

If the output is a TTY, bar is the default. Else, the progress bar will be switched off, except when using --force-progress.

The type `dot' is currently not supported, but wonâ€™t trigger an error to not break wget command lines.

The parameterized types bar:force and bar:force:noscroll will add the effect of --force-progress. These are accepted for better wget compatibility.

--force-progress
Force Wget2 to display the progress bar in any verbosity.

By default, Wget2 only displays the progress bar in verbose mode. One may however, want Wget2 to display the progress bar on screen in conjunction with any other verbosity modes like --no-verbose or --quiet. This is often a desired a property when invoking Wget2 to download several small/large files. In such a case, Wget2 could simply be invoked with this parameter to get a much cleaner output on the screen.

This option will also force the progress bar to be printed to stderr when used alongside the --output-file option.

-N, --timestamping
Turn on time-stamping.

--no-if-modified-since
Do not send If-Modified-Since header in -N mode. Send preliminary HEAD request instead. This has only effect in -N mode.

--no-use-server-timestamps
Donâ€™t set the local fileâ€™s timestamp by the one on the server.

By default, when a file is downloaded, its timestamps are set to match those from the remote file. This allows the use of --timestamping on subsequent invocations of Wget2. However, it is sometimes useful to base the local fileâ€™s timestamp on when it was actually downloaded; for that purpose, the --no-use-server-timestamps option has been provided.

-S, --server-response
Print the response headers sent by HTTP servers.

--spider
When invoked with this option, Wget2 will behave as a Web spider, which means that it will not download the pages, just check that they are there. For example, you can use Wget2 to check your bookmarks:

```
wget2 --spider --force-html -i bookmarks.html
```

This feature needs much more work for Wget2 to get close to the functionality of real web spiders.

-T seconds, --timeout=seconds
Set the network timeout to seconds seconds. This is equivalent to specifying --dns-timeout, --connect-timeout, and --read-timeout, all at the same time.

When interacting with the network, Wget2 can check for timeout and abort the operation if it takes too long. This prevents anomalies like hanging reads and infinite connects. The only timeout enabled by default is a 900-second read timeout. Setting a timeout to 0 disables it altogether. Unless you know what you are doing, it is best not to change the default timeout settings.

All timeout-related options accept decimal values, as well as subsecond values. For example, 0.1 seconds is a legal (though unwise) choice of timeout. Subsecond timeouts are useful for checking server response times or for testing network latency.

--dns-timeout=seconds
Set the DNS lookup timeout to seconds seconds. DNS lookups that donâ€™t complete within the specified time will fail. By default, there is no timeout on DNS lookups, other than that implemented by system libraries.

--connect-timeout=seconds
Set the connect timeout to seconds seconds. TCP connections that take longer to establish will be aborted. By default, there is no connect timeout, other than that implemented by system libraries.

--read-timeout=seconds
Set the read (and write) timeout to seconds seconds. The â€œtimeâ€ of this timeout refers to idle time: if, at any point in the download, no data is received for more than the specified number of seconds, reading fails and the download is restarted. This option does not directly affect the duration of the entire download.

Of course, the remote server may choose to terminate the connection sooner than this option requires. The default read timeout is 900 seconds.

--limit-rate=amount
Limit the download speed to amount bytes per second. Amount may be expressed in bytes, kilobytes with the k suffix, or megabytes with the m suffix. For example, --limit-rate=20k will limit the retrieval rate to 20KB/s. This is useful when, for whatever reason, you donâ€™t want Wget2 to consume the entire available bandwidth.

This option allows the use of decimal numbers, usually in conjunction with power suffixes; for example, --limit-rate=2.5k is a legal value.

Note that Wget2 implements the limiting by sleeping the appropriate amount of time after a network read that took less time than specified by the rate. Eventually this strategy causes the TCP transfer to slow down to approximately the specified rate. However, it may take some time for this balance to be achieved, so donâ€™t be surprised if limiting the rate doesnâ€™t work well with very small files.

-w seconds, --wait=seconds
Wait the specified number of seconds between the retrievals. Use of this option is recommended, as it lightens the server load by making the requests less frequent. Instead of in seconds, the time can be specified in minutes using the â€œmâ€ suffix, in hours using â€œhâ€ suffix, or in days using â€œdâ€ suffix.

Specifying a large value for this option is useful if the network or the destination host is down, so that Wget2 can wait long enough to reasonably expect the network error to be fixed before the retry. The waiting interval specified by this function is influenced by --random-wait, which see.

--waitretry=seconds
If you donâ€™t want Wget2 to wait between every retrieval, but only between retries of failed downloads, you can use this option. Wget2 will use linear backoff, waiting 1 second after the first failure on a given file, then waiting 2 seconds after the second failure on that file, up to the maximum number of seconds you specify.

By default, Wget2 will assume a value of 10 seconds.

--random-wait
Some web sites may perform log analysis to identify retrieval programs such as Wget2 by looking for statistically significant similarities in the time between requests. This option causes the time between requests to vary between 0.5 and 1.5 ### wait seconds, where wait was specified using the --wait option, in order to mask Wget2â€™s presence from such analysis.

A 2001 article in a publication devoted to development on a popular consumer platform provided code to perform this analysis on the fly. Its author suggested blocking at the class C address level to ensure automated retrieval programs were blocked despite changing DHCP-supplied addresses.

The --random-wait option was inspired by this ill-advised recommendation to block many unrelated users from a web site due to the actions of one.

--no-proxy[=exceptions]
If no argument is given, we try to stay backward compatible with Wget1.x and donâ€™t use proxies, even if the appropriate *_proxy environment variable is defined.

If a comma-separated list of exceptions (domains/IPs) is given, these exceptions are accessed without using a proxy. It overrides the `no_proxy' environment variable.

-Q quota, --quota=quota
Specify download quota for automatic retrievals. The value can be specified in bytes (default), kilobytes (with k suffix), or megabytes (with m suffix).

Note that quota will never affect downloading a single file. So if you specify

```
wget2 -Q10k https://example.com/bigfile.gz
```

all of the bigfile.gz will be downloaded. The same goes even when several URLs are specified on the command-line. However, quota is respected when retrieving either recursively, or from an input file. Thus you may safely type

```
wget2 -Q2m -i sites
```

download will be aborted when the quota is exceeded.

Setting quota to 0 or to inf unlimits the download quota.

--restrict-file-names=modes
Change which characters found in remote URLs must be escaped during generation of local filenames. Characters that are restricted by this option are escaped, i.e. replaced with %HH, where HH is the hexadecimal number that corresponds to the restricted character. This option may also be used to force all alphabetical cases to be either lower- or uppercase.

By default, Wget2 escapes the characters that are not valid or safe as part of file names on your operating system, as well as control characters that are typically unprintable. This option is useful for changing these defaults, perhaps because you are downloading to a non-native partition, or because you want to disable escaping of the control characters, or you want to further restrict characters to only those in the ASCII range of values.

The modes are a comma-separated set of text values. The acceptable values are unix, windows, nocontrol, ascii, lowercase, and uppercase. The values unix and windows are mutually exclusive (one will override the other), as are lowercase and uppercase. Those last are special cases, as they do not change the set of characters that would be escaped, but rather force local file paths to be converted either to lower- or uppercase.

When â€œunixâ€ is specified, Wget2 escapes the character / and the control characters in the ranges 0â€"31 and 128â€"159. This is the default on Unix-like operating systems.

When â€œwindowsâ€ is given, Wget2 escapes the characters , |, /, :, ?, â€œ, *, <, >, and the control characters in the ranges 0â€"31 and 128â€"159. In addition to this, Wget2 in Windows mode uses + instead of : to separate host and port in local file names, and uses @ instead of ? to separate the query portion of the file name from the rest. Therefore, a URL that would be saved as www.xemacs.org:4300/search.pl?input=blah in Unix mode would be saved as www.xemacs.org+4300/search.pl@input=blah in Windows mode. This mode is the default on Windows.

If you specify nocontrol, then the escaping of the control characters is also switched off. This option may make sense when you are downloading URLs whose names contain UTF-8 characters, on a system which can save and display filenames in UTF-8 (some possible byte values used in UTF-8 byte sequences fall in the range of values designated by Wget2 as â€œcontrolsâ€).

The ascii mode is used to specify that any bytes whose values are outside the range of ASCII characters (that is, greater than 127) shall be escaped. This can be useful when saving filenames whose encoding does not match the one used locally.

-4, --inet4-only, -6, --inet6-only
Force connecting to IPv4 or IPv6 addresses. With --inet4-only or -4, Wget2 will only connect to IPv4 hosts, ignoring AAAA records in DNS, and refusing to connect to IPv6 addresses specified in URLs. Conversely, with --inet6-only or -6, Wget2 will only connect to IPv6 hosts and ignore A records and IPv4 addresses.

Neither options should be needed normally. By default, an IPv6-aware Wget2 will use the address family specified by the hostâ€™s DNS record. If the DNS responds with both IPv4 and IPv6 addresses, Wget2 will try them in sequence until it finds one it can connect to. (Also see --prefer-family option described below.)

These options can be used to deliberately force the use of IPv4 or IPv6 address families on dual family systems, usually to aid debugging or to deal with broken network configuration. Only one of --inet6-only and --inet4-only may be specified at the same time. Neither option is available in Wget2 compiled without IPv6 support.

--prefer-family=none/IPv4/IPv6
When given a choice of several addresses, connect to the addresses with specified address family first. The address order returned by DNS is used without change by default.

This avoids spurious errors and connect attempts when accessing hosts that resolve to both IPv6 and IPv4 addresses from IPv4 networks. For example, www.kame.net resolves to 2001:200:0:8002:203:47ff:fea5:3085 and to 203.178.141.194. When the preferred family is â€œIPv4â€, the IPv4 address is used first; when the preferred family is â€œIPv6â€, the IPv6 address is used first; if the specified value is â€œnoneâ€, the address order returned by DNS is used without change.

Unlike -4 and -6, this option doesnâ€™t inhibit access to any address family, it only changes the order in which the addresses are accessed. Also note that the reordering performed by this option is stable. It doesnâ€™t affect order of addresses of the same family. That is, the relative order of all IPv4 addresses and of all IPv6 addresses remains intact in all cases.

--tcp-fastopen
Enable support for TCP Fast Open (TFO) (default: on).

TFO reduces connection latency by 1 RT on â€œhotâ€ connections (2nd+ connection to the same host in a certain amount of time).

Currently this works on recent Linux and OSX kernels, on HTTP and HTTPS.

--dns-cache-preload=file
Load a list of IP / Name tuples into the DNS cache.

The format of file is like /etc/hosts: IP-address whitespace Name

This allows to save domain name lookup time, which is a bottleneck in some use cases. Also, the use of HOSTALIASES (which is not portable) can be mimiced by this option.

--dns-cache
Enable DNS caching (default: on).

Normally, Wget2 remembers the IP addresses it looked up from DNS so it doesnâ€™t have to repeatedly contact the DNS server for the same (typically small) set of hosts it retrieves from. This cache exists in memory only; a new Wget2 run will contact DNS again.

However, it has been reported that in some situations it is not desirable to cache host names, even for the duration of a short-running application like Wget2. With --no-dns-cache Wget2 issues a new DNS lookup (more precisely, a new call to â€œgethostbynameâ€ or â€œgetaddrinfoâ€) each time it makes a new connection. Please note that this option will not affect caching that might be performed by the resolving library or by an external caching layer, such as NSCD.

--retry-connrefused
Consider â€œconnection refusedâ€ a transient error and try again. Normally Wget2 gives up on a URL when it is unable to connect to the site because failure to connect is taken as a sign that the server is not running at all and that retries would not help. This option is for mirroring unreliable sites whose servers tend to disappear for short periods of time.

--user=user, --password=password
Specify the username user and password password for HTTP file retrieval. This overrides the lookup of credentials in the .netrc file (--netrc is enabled by default). These parameters can be overridden using the --http-user and --http-password options for HTTP(S) connections.

If neither --http-proxy-user nor --http-proxy-password is given these settings are also taken for proxy authentication.

--ask-password
Prompt for a password on the command line. Overrides the password set by --password (if any).

--use-askpass=command
Prompt for a user and password using the specified command. Overrides the user and/or password set by --user/--password (if any).

--no-iri
Turn off internationalized URI (IRI) support. Use --iri to turn it on. IRI support is activated by default.

You can set the default state of IRI support using the â€œiriâ€ command in .wget2rc. That setting may be overridden from the command line.

--local-encoding=encoding
Force Wget2 to use encoding as the default system encoding. That affects how Wget2 converts URLs specified as arguments from locale to UTF-8 for IRI support.

Wget2 use the function â€œnl_langinfo()â€ and then the â€œCHARSETâ€ environment variable to get the locale. If it fails, ASCII is used.

--remote-encoding=encoding
Force Wget2 to use encoding as the default remote server encoding. That affects how Wget2 converts URIs found in files from remote encoding to UTF-8 during a recursive fetch. This options is only useful for IRI support, for the interpretation of non-ASCII characters.

For HTTP, remote encoding can be found in HTTP â€œContent-Typeâ€ header and in HTML â€œContent-Type http-equivâ€ meta tag.

--input-encoding=encoding
Use the specified encoding for the URLs read from --input-file. The default is the local encoding.

--unlink
Force Wget2 to unlink file instead of clobbering existing file. This option is useful for downloading to the directory with hardlinks.

--cut-url-get-vars
Remove HTTP GET Variables from URLs. For example â€œmain.css?v=123â€ will be changed to â€œmain.cssâ€. Be aware that this may have unintended side effects, for example â€œimage.php?name=sunâ€ will be changed to â€œimage.phpâ€. The cutting happens before adding the URL to the download queue.

--cut-file-get-vars
Remove HTTP GET Variables from filenames. For example â€œmain.css?v=123â€ will be changed to â€œmain.cssâ€.

Be aware that this may have unintended side effects, for example â€œimage.php?name=sunâ€ will be changed to â€œimage.phpâ€. The cutting happens when saving the file, after downloading.

File names obtained from a â€œContent-Dispositionâ€ header are not affected by this setting (see --content-disposition), and can be a solution for this problem.

When --trust-server-names is used, the redirection URL is affected by this setting.

--chunk-size=size
Download large files in multithreaded chunks. This switch specifies the size of the chunks, given in bytes if no other byte multiple unit is specified. By default itâ€™s set on 0/off.

--max-threads=number
Specifies the maximum number of concurrent download threads for a resource. The default is 5 but if you want to allow more or fewer this is the option to use.

-s, --verify-sig[=fail|no-fail]
Enable PGP signature verification (when not prefixed with no-). When enabled Wget2 will attempt to download and verify PGP signatures against their corresponding files. Any file downloaded that has a content type beginning with application/ will cause Wget2 to request the signature for that file.

The name of the signature file is computed by appending the extension to the full path of the file that was just downloaded. The extension used is defined by the --signature-extensions option. If the content type for the signature request is application/pgp-signature, Wget2 will attempt to verify the signature against the original file. By default, if a signature file cannot be found (I.E. the request for it gets a 404 status code) Wget2 will exit with an error code.

This behavior can be tuned using the following arguments: * fail: This is the default, meaning that this is the value when you supply the flag without an argument. Indicates that missing signature files will cause Wget2 to exit with an error code. * no-fail: This value allows missing signature files. A 404 message will still be issued, but the program will exit normally (assuming no unrelated errors).

Additionally, --no-verify-sig disables signature checking altogether --no-verify-sig does not allow any arguments.

--signature-extensions
Specify the file extensions for signature files, without the leading â€œ.â€. You may specify multiple extensions as a comma separated list. All the provided extensions will be tried simultaneously when looking for the signature file. The default is â€œsigâ€.

--gnupg-homedir
Specifies the gnupg home directory to use when verifying PGP signatures on downloaded files. The default for this is your systemâ€™s default home directory.

--verify-save-failed
Instructs Wget2 to keep files that donâ€™t pass PGP signature validation. The default is to delete files that fail validation.

--xattr
Saves documents metadata as â€œuser POSIX Extended Attributesâ€ (default: on). This feature only works if the file system supports it. More info on https://freedesktop.org/wiki/CommonExtendedAttributes.

Wget2 currently sets * user.xdg.origin.url * user.xdg.referrer.url * user.mime_type * user.charset

To display the extended attributes of a file (Linux): getfattr -d

--metalink
Follow/process metalink URLs without saving them (default: on).

Metalink files describe downloads incl. mirrors, files, checksums, signatures. This allows chunked downloads, automatically taking the nearest mirrors, preferring the fastest mirrors and checking the download for integrity.

--fsync-policy
Enables disk syncing after each write (default: off).

--http2-request-window=number
Set max. number of parallel streams per HTTP/2 connection (default: 30).

--keep-extension
This option changes the behavior for creating a unique filename if a file already exists.

The standard (default) pattern for file names is ., the new pattern is _..

The idea is to use such files without renaming when the use depends on the extension, like on Windows.

This option doesn not change the behavior of --backups.

Directory Options

-nd, --no-directories
Do not create a hierarchy of directories when retrieving recursively. With this option turned on, all files will get saved to the current directory, without clobbering (if a name shows up more than once, the filenames will get extensions .n).

-x, --force-directories
The opposite of -nd: create a hierarchy of directories, even if one would not have been created otherwise. E.g. wget2 -x https://example.com/robots.txt will save the downloaded file to example.com/robots.txt.

-nH, --no-host-directories
Disable generation of host-prefixed directories. By default, invoking Wget2 with -r https://example.com/ will create a structure of directories beginning with example.com/. This option disables such

behavior.

--protocol-directories

Use the protocol name as a directory component of local file names. For example, with this option, wget2 -r https://example.com will save to https/example.com/... rather than just to example.com/....

--cut-dirs=number

Ignore a number of directory components. This is useful for getting a fine-grained control over the directory where recursive retrieval will be saved.

Take, for example, the directory at https://example.com/pub/sub/. If you retrieve it with -r, it will be saved locally under example.com/pub/sub/. While the -nH option can remove the example.com/ part, you are still stuck with pub/sub/. This is where --cut-dirs comes in handy; it makes Wget2 not â€œseeâ€ a number of remote directory components. Here are several examples of how --cut-dirs option works. No options -> example.com/pub/sub/ --cut-dirs=1 -> example.com/sub/ --cut-dirs=2 -> example.com/ -nH -> pub/sub/ -nH --cut-dirs=1 -> sub/ -nH --cut-dirs=2 -> . If you just want to get rid of the directory structure, this option is similar to a combination of -nd and -P. However, unlike -nd, --cut-dirs does not lose with subdirectories. For instance, with -nH --cut-dirs=1, a beta/ subdirectory will be placed to sub/beta/, as one would expect.

-P prefix, --directory-prefix=prefix

Set directory prefix to prefix. The directory prefix is the directory where all other files and subdirectories will be saved to, i.e. the top of the retrieval tree. The default is ., the current directory. If the directory prefix doesnâ€™t exist, it will be created.

HTTP Options

--default-page=name

Use name as the default file name when it isnâ€™t known (i.e., for URLs that end in a slash), instead of index.html.

--default-http-port=port

Set the default port for HTTP URLs (default: 80).

This is mainly for testing purposes.

--default-https-port=port

Set the default port for HTTPS URLs (default: 443).

This is mainly for testing purposes.

-E, --adjust-extension

If a file of type application/xhtml+xml or text/html is downloaded and the URL does not end with the regexp \.[Hh][Tt][Mm][Ll]?, this option will cause the suffix .html to be appended to the local filename. This is useful, for instance, when youâ€™re mirroring a remote site that uses .asp pages, but you want the mirrored pages to be viewable on your stock Apache server. Another good use for this is when youâ€™re downloading CGI-generated materials. A URL like https://example.com/article.cgi?25 will be saved as article.cgi?25.html.

Note that filenames changed in this way will be re-downloaded every time you re-mirror a site, because Wget2 canâ€™t tell that the local X.html file corresponds to remote URL X (since it doesnâ€™t yet know that the URL produces output of type text/html or application/xhtml+xml.

Wget2 will also ensure that any downloaded files of type text/css end in the suffix .css.

At some point in the future, this option may well be expanded to include suffixes for other types of content, including content types that are not parsed by Wget.

--http-user=user, --http-password=password

Specify the user and password for HTTP authentication. According to the type of the challenge, Wget will encode them using either the â€œbasicâ€ (insecure), the â€œdigestâ€, or the Windows â€œNTLMâ€ authentication scheme.

If possible, put your credentials into ~/.netrc (see also --netrc and --netrc-file options) or into .wget2rc. This is far more secure than using the command line which can be seen by any other user. If the passwords are really important, do not leave them lying in those files either. Edit the files and delete them after Wget2 has started the download.

In ~/.netrc passwords may be double quoted to allow spaces. Also, escape characters with a backslash if needed. A backslash in a password always needs to be escaped, so use \\ instead of a single \.

Also see --use-askpass and --ask-password for an interactive method to provide your password.

--http-proxy-user=user, --http-proxy-password=password

Specify the user and password for HTTP proxy authentication. See --http-user for details.

--http-proxy=proxies

Set comma-separated list of HTTP proxies. The environment variable `http_proxy' will be overridden.

Exceptions can be set via the environment variable `no_proxy' or via --no-proxy.

--https-proxy=proxies

Set comma-separated list of HTTPS proxies. The environment variable `https_proxy' will be overridden.

Exceptions can be set via the environment variable `no_proxy' or via --no-proxy.

--no-http-keep-alive

Turn off the â€œkeep-aliveâ€ feature for HTTP(S) downloads. Normally, Wget2 asks the server to keep the connection open so that, when you download more than one document from the same server, they get transferred over the same TCP connection. This saves time and at the same time reduces the load on the server.

This option is useful when, for some reason, persistent (keep-alive) connections donâ€™t work for you, for example due to a server bug or due to the inability of server-side scripts to cope with the connections.

--no-cache

Disable server-side cache. In this case, Wget2 will send the remote server appropriate directives (Cache-Control: no- cache and Pragma: no-cache) to get the file from the remote service, rather than returning the cached version. This is especially useful for retrieving and flushing out-of-date documents on proxy servers.

Caching is allowed by default.

--no-cookies

Disable the use of cookies. Cookies are a mechanism for maintaining server-side state. The server sends the client a cookie using the â€œSet-Cookieâ€ header, and the client responds with the same cookie upon further requests. Since cookies allow the server owners to keep track of visitors and for sites to exchange this information, some consider them a breach of privacy. The default is to use cookies; however, storing cookies is not on by default.

--load-cookies file

Load cookies from file before the first HTTP(S) retrieval. file is a textual file in the format originally used by Netscapeâ€™s cookies.txt file.

You will typically use this option when mirroring sites that require that you be logged in to access some or all of their content. The login process typically works by the web server issuing an HTTP cookie upon receiving and verifying your credentials. The cookie is then resent by the browser when accessing that part of the site, and so proves your identity.

Mirroring such a site requires Wget2 to send the same cookies your browser sends when communicating with the site. This is achieved by --load-cookies: simply point Wget2 to the location of the cookies.txt file, and it will send the same cookies your browser would send in the same situation. Different browsers keep textual cookie files in different locations:

â€œNetscape 4.x.â€ The cookies are in ~/.netscape/cookies.txt.

â€œMozilla and Netscape 6.x.â€ Mozillaâ€™s cookie file is also named cookies.txt, located somewhere under ~/.mozilla, in the directory of your profile. The full path usually ends up looking somewhat like ~/.mozilla/default/some-weird- string/cookies.txt.

â€œInternet Explorer.â€ You can produce a cookie file Wget2 can use by using the File menu, Import and Export, Export Cookies. This has been tested with Internet Explorer 5; it is not guaranteed to work with earlier versions.

â€œOther browsers.â€ If you are using a different browser to create your cookies, --load-cookies will only work if you can locate or produce a cookie file in the Netscape format that Wget2 expects.

If you cannot use --load-cookies, there might still be an alternative. If your browser supports a â€œcookie managerâ€, you can use it to view the cookies used when accessing the site youâ€™re mirroring. Write down the name and value of the cookie, and manually instruct Wget2 to send those cookies, bypassing the â€œofficialâ€ cookie support:

```
wget2 --no-cookies --header "Cookie: ="
```

--save-cookies file

Save cookies to file before exiting. This will not save cookies that have expired or that have no expiry time (so-called â€œsession cookiesâ€), but also see --keep-session-cookies.

--keep-session-cookies

When specified, causes --save-cookies to also save session cookies. Session cookies are normally not saved because they are meant to be kept in memory and forgotten when you exit the browser. Saving them is useful on sites that require you to log in or to visit the home page before you can access some pages. With this option, multiple Wget2 runs are considered a single browser session as far as the site is concerned.

Since the cookie file format does not normally carry session cookies, Wget2 marks them with an expiry timestamp of 0. Wget2â€™s --load-cookies recognizes those as session cookies, but it might confuse other browsers. Also note that cookies so loaded will be treated as other session cookies, which means that if you want --save-cookies to preserve them again, you must use --keep-session-cookies again.

--cookie-suffixes=file

Load the public suffixes used for cookie checking from the given file.

Normally, the underlying libpsl loads this data from a system file or it has the data built in. In some cases you might want to load an updated PSL, e.g. from https://publicsuffix.org/list/public_suffix_list.dat.

The PSL allows to prevent setting of â€œsuper-cookiesâ€ that lead to cookie privacy leakage. More details can be found on https://publicsuffix.org/.

--ignore-length

Unfortunately, some HTTP servers (CGI programs, to be more precise) send out bogus â€œContent-Lengthâ€ headers, which makes Wget2 go wild, as it thinks not all the document was retrieved. You can spot this syndrome if Wget retries getting the same document again and again, each time claiming that the (otherwise normal) connection has closed on the very same byte.

With this option, Wget2 will ignore the â€œContent-Lengthâ€ header as if it never existed.

--header=header-line

Send header-line along with the rest of the headers in each HTTP request. The supplied header is sent as-is, which means it must contain name and value separated by colon, and must not contain newlines.

You may define more than one additional header by specifying --header more than once.

```
wget2 --header='Accept-Charset: iso-8859-2' \
      --header='Accept-Language: hr'        \
        https://example.com/
```

Specification of an empty string as the header value will clear all previous user-defined headers.

This option can be used to override headers otherwise generated automatically. This example instructs Wget2 to connect to localhost, but to specify example.com in the â€œHostâ€ header:

```
wget2 --header="Host: example.com" http://localhost/
```

--max-redirect=number

Specifies the maximum number of redirections to follow for a resource. The default is 20, which is usually far more than necessary. However, on those occasions where you want to allow more (or fewer), this is the option to use.

--proxy-user=user, --proxy-password=password [Not implemented, use --http-proxy-password]

Specify the username user and password password for authentication on a proxy server. Wget2 will encode them using the â€œbasicâ€ authentication scheme.

Security considerations similar to those with --http-password pertain here as well.

--referer=url

Include `Referer: urlâ€™ header in HTTP request. Useful for retrieving documents with server-side processing that assume they are always being retrieved by interactive web browsers and only come out properly when Referer is set to one of the pages that point to them.

--save-headers

Save the headers sent by the HTTP server to the file, preceding the actual contents, with an empty line as the separator.

-U agent-string, --user-agent=agent-string

Identify as agent-string to the HTTP server.

The HTTP protocol allows the clients to identify themselves using a â€œUser-Agentâ€ header field. This enables distinguishing the WWW software, usually for statistical purposes or for tracing of protocol violations. Wget normally identifies as Wget/version, version being the current version number of Wget.

However, some sites have been known to impose the policy of tailoring the output according to the â€œUser-Agentâ€-supplied information. While this is not such a bad idea in theory, it has been abused by servers denying information to clients other than (historically) Netscape or, more frequently, Microsoft Internet Explorer. This option allows you to change the â€œUser-Agentâ€ line issued by Wget. Use of this option is discouraged, unless you really know what you are doing.

Specifying empty user agent with --user-agent="" instructs Wget2 not to send the â€œUser-Agentâ€ header in HTTP requests.

--post-data=string, --post-file=file

Use POST as the method for all HTTP requests and send the specified data in the request body. â€"post-data sends string as data, whereas --post-file sends the contents of file. Other than that, they work in exactly the same way. In particular, they both expect content of the form â€œkey1=value1&key2=value2â€, with percent-encoding for special characters; the only difference is that one expects its content as a command-line parameter and the other accepts its content from a file. In particular, --post-file is not for transmitting files as form attachments: those must appear as â€œkey=valueâ€ data (with appropriate percent-coding) just like everything else. Wget2 does not currently support â€œmultipart/form-dataâ€ for transmitting POST data; only â€œapplication/x-www-form-urlencodedâ€. Only one of --post-data and --post-file should be specified.

Please note that wget2 does not require the content to be of the form â€œkey1=value1&key2=value2â€, and neither does it test for it. Wget2 will simply transmit whatever data is provided to it. Most servers however expect the POST data to be in the above format when processing HTML Forms.

When sending a POST request using the --post-file option, Wget2 treats the file as a binary file and will send every character in the POST request without stripping trailing newline or formfeed characters. Any other control characters in the text will also be sent as-is in the POST request.

Please be aware that Wget2 needs to know the size of the POST data in advance. Therefore the argument to --post-file must be a regular file; specifying a FIFO or something like /dev/stdin wonâ€™t work. Itâ€™s not quite clear how to work around this limitation inherent in HTTP/1.0. Although HTTP/1.1 introduces chunked transfer that doesnâ€™t require knowing the request length in advance, a client canâ€™t use chunked unless it knows itâ€™s talking to an HTTP/1.1 server. And it canâ€™t know that until it receives a response, which in turn requires the request to have been completed â€" a chicken-and-egg problem.

If Wget2 is redirected after the POST request is completed, its behaviour depends on the response code returned by the server. In case of a 301 Moved Permanently, 302 Moved Temporarily or 307 Temporary Redirect, Wget2 will, in accordance with RFC2616, continue to send a POST request. In case a server wants the client to change the Request method upon redirection, it should send a 303 See Other response code.

This example shows how to log in to a server using POST and then proceed to download the desired pages, presumably only accessible to authorized users:

```
# Log in to the server.  This can be done only once.
wget2 --save-cookies cookies.txt \
     --post-data  'user=foo&password=bar' \
     http://example.com/auth.php

# Now grab the page or pages we care about.
wget2 --load-cookies cookies.txt \
     -p http://example.com/interesting/article.php
```

If the server is using session cookies to track user authentication, the above will not work because --save-cookies will not save them (and neither will browsers) and the cookies.txt file will be empty. In that case use --keep-session-cookies along with --save-cookies to force saving of session cookies.

--method=HTTP-Method

For the purpose of RESTful scripting, Wget2 allows sending of other HTTP Methods without the need to explicitly set them using --header=Header-Line. Wget2 will use whatever string is passed to it after --method as the HTTP Method to the server.

--body-data=Data-String, --body-file=Data-File

Must be set when additional data needs to be sent to the server along with the Method specified using --method. --body-data sends string as data, whereas --body-file sends the contents of file. Other than that, they work in exactly the same way.

Currently, --body-file is not for transmitting files as a whole. Wget2 does not currently support â€œmultipart/form-dataâ€ for transmitting data; only â€œapplication/x-www-form-urlencodedâ€. In the future, this may be changed so that wget2 sends the --body-file as a complete file instead of sending its contents to the server. Please be aware that Wget2 needs to know the contents of BODY Data in advance, and hence the argument to --body-file should be a regular file. See --post-file for a more detailed explanation. Only one of --body-data and --body-file should be specified.

If Wget2 is redirected after the request is completed, Wget2 will suspend the current method and send a GET request till the redirection is completed. This is true for all redirection response codes except 307 Temporary Redirect which is used to explicitly specify that the request method should not change. Another exception is when the method is set to â€œPOSTâ€, in which case the redirection rules specified under --post-data are followed.

--content-disposition

If this is set to on, experimental (not fully-functional) support for â€œContent-Dispositionâ€ headers is enabled. This can currently result in extra round-trips to the server for a â€œHEADâ€ request, and is known to suffer from a few bugs, which is why it is not currently enabled by default.

This option is useful for some file-downloading CGI programs that use â€œContent-Dispositionâ€ headers to describe what the name of a downloaded file should be.

--content-on-error

If this is set to on, wget2 will not skip the content when the server responds with a http status code that indicates error.

--save-content-on

This takes a comma-separated list of HTTP status codes to save the content for.

You can use â€™*â€™ for ANY. An exclamation mark (!) in front of a code means `exception'.

Example 1: --save-content-on="*,!404" would save the content on any HTTP status, except for 404.

Example 2: --save-content-on=404 would save the content only on HTTP status 404.

The older --content-on-error behaves like --save-content-on=*.

--trust-server-names

If this is set to on, on a redirect the last component of the redirection URL will be used as the local file name. By default it is used the last component in the original URL.

--auth-no-challenge

If this option is given, Wget2 will send Basic HTTP authentication information (plaintext username and password) for all requests.

Use of this option is not recommended, and is intended only to support some few obscure servers, which never send HTTP authentication challenges, but accept unsolicited auth info, say, in addition to form-based authentication.

--compression=TYPE

If this TYPE(identity, gzip, deflate, xz, lzma, br, bzip2, zstd, lzip or any combination of it) is given, Wget2 will set â€œAccept-Encodingâ€ header accordingly. --no-compression means no â€œAccept-Encodingâ€ header at all. To set â€œAccept-Encodingâ€ to a custom value, use --no-compression in combination with --header="Accept-Encoding: xxx".

Compatibility-Note: none type in Wget 1.X has the same meaning as identity type in Wget2.

--download-attr=[strippath|usepath]

The download HTML5 attribute may specify (or better: suggest) a file name for the href URL in a and area tags. This option tells Wget2 to make use of this file name when saving. The two possible values are `strippath' to strip the path from the file name. This is the default.

The value `usepath' takes the file name as as including the directory. This is very dangerous and we canâ€™t stress enough not to use it on untrusted input or servers ! Only use this if you really trust the input or the server.

HTTPS (SSL/TLS) Options

To support encrypted HTTP (HTTPS) downloads, Wget2 must be compiled with an external SSL library. The current default is GnuTLS. In addition, Wget2 also supports HSTS (HTTP Strict Transport Security). If Wget2 is compiled without SSL support, none of these options are available.

--secure-protocol=protocol

Choose the secure protocol to be used (default: auto).

Legal values are auto, SSLv3, TLSv1, TLSv1_1, TLSv1_2, TLSv1_3 and PFS.

If auto is used, the TLS libraryâ€™s default is used.

Specifying SSLv3 forces the use of the SSL3. This is useful when talking to old and buggy SSL server implementations that make it hard for the underlying TLS library to choose the correct protocol version.

Specifying PFS enforces the use of the so-called Perfect Forward Security cipher suites. In short, PFS adds security by creating a one-time key for each TLS connection. It has a bit more CPU impact on client and server. We use known to be secure ciphers (e.g. no MD4) and the TLS protocol.

TLSv1 enables TLS1.0 or higher. TLSv1_1 enables TLS1.1 or higher. TLSv1_2 enables TLS1.2 or higher. TLSv1_3 enables TLS1.3 or higher.

Any other protocol string is directly given to the TLS library, currently GnuTLS, as a â€œpriorityâ€ or â€œcipherâ€ string. This is for users who know what they are doing.

--https-only

When in recursive mode, only HTTPS links are followed.

--no-check-certificate

Donâ€™t check the server certificate against the available certificate authorities. Also donâ€™t require the URL host name to match the common name presented by the certificate.

The default is to verify the serverâ€™s certificate against the recognized certificate authorities, breaking the SSL handshake and aborting the download if the verification fails. Although this provides more secure downloads, it does break interoperability with some sites that worked with previous Wget versions, particularly those using self-signed, expired, or otherwise invalid certificates. This option forces an â€œinsecureâ€ mode of operation that turns the certificate verification errors into warnings and allows you to proceed.

If you encounter â€œcertificate verificationâ€ errors or ones saying that â€œcommon name doesnâ€™t match requested host nameâ€, you can use this option to bypass the verification and proceed with the download. Only use this option if you are otherwise convinced of the siteâ€™s authenticity, or if you really donâ€™t care about the validity of its certificate. It is almost always a bad idea not to check the certificates when transmitting confidential or important data. For self-signed/internal certificates, you should download the certificate and verify against that instead of forcing this insecure mode. If you are really sure of not desiring any certificate verification, you can specify --check-certificate=quiet to tell Wget2 to not print any warning about invalid certificates, albeit in most cases this is the wrong thing to do.

--certificate=file

Use the client certificate stored in file. This is needed for servers

```
--certificate=type
       Specify the type of the client certificate.  Legal values are PEM
       (assumed by default) and DER, also known as ASN1.

--private-key=file
       Read the private key from file.  This allows you to provide the private
       key in a file separate from the certificate.

--private-key-type=type
       Specify the type of the private key.  Accepted values are PEM (the
       default) and DER.

--ca-certificate=file
       Use the CA bundle file with the bundle of certificate authorities (aka"CAs")
       to verify the peers.  The certificate must be in PEM format.

       Without this option Wget looks for CA certificates at the system-
       specified locations, chosen at OpenSSL installation time.

--ca-directory=directory
       Specifies directory containing CA certificates in PEM format.  Each
       file contains one CA certificate, and the file name is based on a hash
       value derived from the certificate.  This is achieved by processing a
       certificate directory with the "c_rehash" utility supplied with
       OpenSSL.  Using certificate directory is more efficient than loading
       when many certificates are installed because it allows Wget to fetch
       certificates on demand.

       Without this option Wget looks for CA certificates at the system-
       specified locations, chosen at OpenSSL installation time.

--crl-file=file
       Specifies a CRL file in file.  This is needed for certificates that
       have been revoked by the CAs.

--random-file=file
       [OpenSSL and LibreSSL only] the file as the source of random data for
       seeding the pseudo-random number generator on systems without
       /dev/random.

       On such systems the SSL library needs an external source of randomness
       to initialize. Randomness may be provided by the EGD (see, e.g. prngd
       or new) or read from an external source specified by the user.  If this option
       is not specified, Wget looks for random data in $RANDFILE or, if that
       is unset, in $HOME/.rnd.

       If "OpenSSL" is being the"OpenSSL and OpenSSH PRNG: disabling SSL.ML"
       error, you should provide random data using some of the methods
       described above.

--egd-file=file
       [OpenSSL only] the file as the EGD socket.  Not needed on systems
       with the OpenSSL library compiled to read from EGD socket by default or
       with unpredictable system sources and makes it available to other programs
       that might need it.  Encryption software, such as the SSL library,
       needs sources of non-predictable random data to seed the random
       generator used to produce cryptographically strong keys.

       OpenSSL allows the user to specify his own source of entropy using the
       "RAND_EGD_FILENAME" environment variable.  If this variable is unset, or if the
       specified file does not produce enough randomness, OpenSSL will read
       random data from EGD socket specified using this option.

       If this option is not specified (and the equivalent startup command is
       not used), EGD is never contacted.  EGD is not needed on modern Unix
       systems that support /dev/random.

--hsts
       Wget supports HSTS (HTTP Strict Transport Security, RFC 6797) by
       default. It can be disabled with "--no-hsts". Conversions to HTTPS on
       the fly, as a consequence, Wget would ignore all the "Address(Protocol)
       Security"HSTS headers, and would not enforce any existing HSTS policy.

--hsts-file=file
       By default, Wget stores its HSTS data in $XDG_DATA_HOME/wget/wget-
       hsts or, if $XDG_DATA_HOME is not set, in ~/.local/wget/wget-hsts.  You
       can use "--hsts-file" to override this.

       Wget will use the supplied file as the HSTS database.  Such file must
       conform to the correct HSTS database format used by Wget.  If Wget
       cannot parse the provided file, the behaviour is unspecified.

       To disable persistent storage use "--no-hsts-file.

       Each line contains an HSTS entry, i.e. a line that has ieee 1003.1-2008 Base Definitions, Chapter 3 Definitions
       HSTS entry (ie. a site that has issued a "Address(Protocol)Security"
       header in the past, and which has not expired yet). Lines starting
       with a pound ("#"), and empty lines are ignored by Wget.
       Please note that in spite of this convenient human-readability hand-
       hacking the HSTS database is generally not a good idea.

       An HSTS entry line consists of several fields separated by one or more
       whitespaces:

            ... ... ... ...

       The hostname and port fields indicate the hostname and port to which
       the given HSTS policy applies. The port field may be zero, and it
       will, in most of the cases. That means that the port number will not
       be taken into account when deciding whether such HSTS policy should be
       enforced on a URL or not. Strictly speaking, a given HSTS policy with
       port 0 (different to zero, both the target hostnames port and port will
       be evaluated and the HSTS policy will only be applied if both of them
       match. This feature has been included for testing/development purposes
       only. The Wget testsuite (in testsuite) creates HSTS databases with
       explicit ports to ease testing the HSTS feature in "localhost" servers.
       Applying HSTS policies to ports other than the default ones is
       discouraged by RFC 6797 (see Appendix A "Implementation Notes" and
       Appendix B "Deployment Issues" for a detailed explanation of the risks
       be used in production environments and port will typically be zero
       The last three fields do what they are expected to. The field
       "Address(Protocol)Subdomains" can be 1 or 0 depending on whether
       subdomains of the target domain should be part of the HSTS policy
       as well. The created and max-age fields hold the timestamp values of
       when the HSTS policy was created (last time parsed by Wget) and its
       value "max-age", which states how long should the HSTS policy remain
       active, measured in seconds elapsed since the timestamp stored in
       created. Once that time has passed, that HSTS policy will no longer be
       valid and will eventually be removed from the database.

       If you supply your own HSTS database via "--hsts-file", be aware that
       Wget may modify the provided file if any change occurs between the
       HSTS policies requested by visited hosts and the entries stored in
       this file. When Wget exits, it effectively updates ~/.local/wget/wget-hsts.
       This behaviour applies to the default database file (~/.wget-hsts) as well:
       it will not be created until some server enforces an HSTS policy.

       Care is taken not to override possible changes made by other Wget
       processes at the same time over the HSTS database. Before dumping the
       updated HSTS entries on the file, Wget will re-read it and merge the
       changes.

       Using a custom HSTS database and/or modifying an existing one is
       discouraged. For more information about the possible attacks, threats
       arises from such practice, see section 14 "Address(Security Considerations)" of
       RFC 6797, specially section 14.9 "Address(Persistent Manipulation of HSTS Policy
       Stores)".

--hsts-preload
       Enable loading of a HSTS Preload list as supported by libhsts.
       (default: on, if built with libhsts).

--hsts-preload-file=file
       If built with libhsts, Wget uses the HSTS data provided by the
       preload file to enforce HTTPS connections. This preload distribution or if
       you want to load your own file, use this option.

       The data file must be in DNSSA format as generated by libhsts"" tool
       hsts-make-dafsa.

--hpkp
       Enable HTTP Public Key Pinning (HPKP) (default: on).

       This is a Trust On First Use (TOFU) mechanism to add another security
       layer to HTTPS (RFC 7469).

       The certificate key data of a previously established TLS session will
       be compared with the current data.  In case both doesnâ€™t match, the
       connection will be terminated.

--hpkp-file=file
       By default, Wget stores its HPKP data in $XDG_DATA_HOME/wget/wget-
       hpkp or, if $XDG_DATA_HOME is not set, in ~/.local/wget/wget-hpkp.  You
       can use "--hpkp-file" to override this.

       Wget will use the supplied file as the HPKP database.  Such file must
       conform to the correct HPKP database format used by Wget.  If Wget2
       cannot parse the provided file, the behaviour is unspecified.

       To disable persistent storage use "--no-hpkp-file.

--tls-resume
       Enable TLS Session Resumption which is disabled as default.

       For TLS Session Resumption the session data of a previously established
       TLS session is needed.

       There are several security flaws related to TLS 1.2 session resumption
       which are explained in detail at:
       https://web.archive.org/web/20171013213045/https://blog.filippo.io/we-
       need-to-talk-about-session-tickets/

--tls-session-file=file
       Wget2 will use the session file as TLS Session data. In
       $XDG_DATA_HOME/wget/wget-session or, if $XDG_DATA_HOME is not set, in
       ~/.local/wget/wget-session.  You can use "--tls-session-file to
       override this.

       Wget2 will use the supplied file as the TLS Session database.  Such
       file must conform to the correct TLS Session database format used by
       Wget.  If Wget2 cannot parse the provided file, the behaviour is
       unspecified.

       To disable persistent storage use "--no-tls-session-file.

--tls-false-start
       Enable TLS False start (default: on).

       This reduces TLS negotiation by one RT and thus speeds up HTTPS
       connections.

       More details at https://tools.ietf.org/html/rfc7918.

--check-hostname
       Enable TLS SNI verification (default: on).

--ocsp
       Enable OCSP server access to check the possible revocation the HTTPS
       server certificates(default: on).

       This procedure is pretty slow (connect to server, HTTP request,
       response). Wget2 supports pipelining, so it sends the server ocsp OCSP response
       within TLS handshake) and persistent OCSP caching.

--ocsp-date
       Check if OCSP response is too old. (default: on)

--ocsp-nonce
       Allow nonce checking when verifying OCSP responses. (default: on)

--ocsp-server
       Set OCSP server address (default: OCSP server given in certificate).

--ocsp-stapling
       Enable support for OCSP stapling (default: on).

--ocsp-file=file
       By default, Wget2 stores its TLS Session data in
       $XDG_DATA_HOME/wget/wget-ocsp or, if $XDG_DATA_HOME is not set, in
       ~/.local/wget/wget-ocsp.  You can use "--ocsp-file to override this.

       Wget2 will use the supplied file as the OCSP database.  Such file must
       conform to the correct OCSP database format used by Wget.  If Wget2
       cannot parse the provided file, the behaviour is unspecified.

       To disable persistent OCSP caching use "--no-ocsp-file.

--dane (experimental)
       Enable DANE certificate verification (default: off).

       In case the server verification fails due to missing CA certificates
       (e.g. empty certification pool), this option enables checking the TLSA
       DNS entries via DNS.

       You should have DNSSEC set up to avoid MITM attacks.  Also, the
       destination hostâ€™s DNS entries need to be set up DANE.

       Wget2 requests HTTP/2 via ALPN.  If available it is preferred over
       HTTP/1.1.  Up to 20 streams are used in parallel within a single
       connection.

--http2-only
       Resist on using HTTP/2 and error if a server doesnâ€™t accept it.  This
       is mainly for testing.

--https-enforce=mode
       Ask how to deal with URLs that are not explicitly HTTPS (where scheme
       isn'â€™t https://) (default: none)

mode=none
       Use HTTP for URLs without scheme.  In recursive operation the scheme of
       the parent document is taken as default.

mode=soft
       Use HTTPS first when the scheme is HTTP or not given.  On failure fall
       back to HTTP.

mode=hard
       Only use HTTPS, no matter if a HTTP scheme is given or not.  Do not
       fall back to HTTPS.

Recursive Retrieval Options
'-r, --recursive
       Turn on recursive retrieving.  The default maximum depth is 5.

-l depth, --level=depth
       Specify recursion maximum depth level depth.

--delete-after
       This option tells Wget2 to delete every single file it downloads, after
       having done so.  It is useful for pre- fetching popular pages through a
       proxy, e.g.:

            wget2 -r -nd --delete-after https://example.com/~popular/page/

       The -r option is to retrieve recursively, and -nd to not create
       directories.

       Note that when DeâˆÑelete-after is specified, --convert-links is ignored,
       so -nc/-N files are simply not created in the first place.

-k, --convert-links
       After the download is complete, convert the links in the document to
       make them suitable for local viewing.  This affects not only the
       visible hyperlinks, but any part of the document that links to external
       content, such as embedded images, links to style sheets, hyperlinks to
       non-HTML content, etc.

       Each link will be changed in one of the two ways:

       1. The links to files that have been downloaded by Wget2 will be
          changed to refer to the file they point to as a relative link.

          Example: if the downloaded file /foo/doc.html links to
          /bar/img.gif, also downloaded, then the link in doc.html will be
          modified to point to ../bar/img.gif. This kind of transformation
          works reliably for arbitrary combinations of directories.

       2. The links to files that have not been downloaded by Wget2 will be
          changed to include host name and absolute path of the location they
          point to.

          Example: if the downloaded file /foo/doc.html links to /bar/img.gif
          (or to ../bar/img.gif), then the link in doc.html will be modified
          to point to https://example.com/bar/img.gif.

       Because of this, local browsing works reliably: if a linked file was
       downloaded, the link will refer to its local name; if it was not
       downloaded, the link will refer to its full Internet address rather
       than presenting a broken link.  The fact that the former links are
       converted to relative ones ensures that you can move the downloaded
       hierarchy to another directory.

       Note that only at the end of the download can Wget2 know which links
       have been downloaded.  Because of that, the work done by -k will be
       performed at the end of all the downloads.

--convert-file-only
       This option converts only the filename part of the URLs, leaving the
       rest of the URLs untouched.  This filename part is sometimes referred
       to as the bNbasenameâ€Ñ, although we avoid that term here in order not to
       cause confusion.

       It works particularly well in conjunction with --adjust-extension,
       although this coupling is not enforced.  It proves useful to populate
       Internet caches with files downloaded from different hosts.

       Examples if some link points to //foo.com/bar.cgi?xyz with --adjust-
       extension asserted and its local destination is intended to be
       ./foo.com/bar.cgi?xyz.css, then the link would be converted to
       //foo.com/bar.cgi?xyz.css.  Note that only the filename part has been
       modified.  The rest of the URL has been left untouched, including the
       net path ("//") which would otherwise be converted to "http://" by Wget2
       converted to the effective scheme (ie. bNâ€Ñtpsâ€Ñ).

-K, --backup-converted
       When converting a file, back up the original version with a .orig
       suffix.  Affects the behaviour of -N.

-m, --mirror
       Turn on options suitable for mirroring.  This option turns on recursion
       and time-stamping, sets infinite recursion depth.  It is currently
       equivalent to -r -N -l inf.

-p, --page-requisites
       This option causes Wget2 to download all the files that are necessary
       to properly display a given HTML page.  This includes such things as
       inlined images, sounds, and referenced stylesheets.

       Ordinarily, when downloading a single HTML page, any requisite
       documents that may be needed to display it properly are not downloaded.
       Using -r together with -l can help, but since Wget2 does not ordinarily
       distinguish external and inlined documents, we are therefore left with
       bNot enough documentâ€Ñ that are missing their requisites.

       For instance, say document 1.html contains an "img referencing
       1.gif and an "A" pointing to external document 2.html.  Say that
       2.html is similar but that its image is 2.gif and it links to 3.html.
       Say this continues up to some arbitrarily high number.

       If one executes the command:

            wget2 -r -l 2 http://1.html

       then 1.html, 1.gif, 2.html, 2.gif, and 3.html will be downloaded.  As
       you can see, 3.html is without its requisite 3.gif because Wget2 is
       simply counting the number of hops (up to 2 away from 1.html) in order
       to determine where to stop the recursion.  However, with this command:

            wget2 -r -l 2 -p http://1.html

       all the above files and 3.html&#39;s requisite 3.gif will be downloaded.
       Similarly,

            wget2 -r -l 1 -p https://1.html

       will cause 1.html, 1.gif, 2.html, and 2.gif to be downloaded.  One
       might think that:

            wget2 -r -l 0 -p https://1.html

       would download just 1.html and 1.gif, but unfortunately this is not the
       case, because -l 0 is equivalent to -l inf, that is, infinite
       recursion.  To download a single HTML page (or a handful of them, all
       specified on the commandline or in a -i URL input file) and its (or
       them) requisites, simply leave off -r and -l:

            wget2 -p https://1.html

       Note that Wget2 will behave as if -r had been specified, but only that
       single page and its requisites will be downloaded.  Links from that
       page to external documents will not be followed.  Actually, to download
       a single page and all its requisites (even if they exist on separate
       websites), and make sure the lots of pages displayed properly locally, this author
       likes to use a few options in addition to -p:

            wget2 -E -H -k -K -p https://

       To finish off this topic, itâ€™s worth knowing that Wget2â€™s idea of an
       external document link is any URL specified in an "A", an
       "AREA", an "LINK" or a "SCRIPT".

--strict-comments
       Obsolete option for compatibility with Wget1.x.  Wget2 always
       terminates comments at the first occurrence of -->, as popular browsers
       do.

--robots
       Enable the Robots Exclusion Standard (default: on).

       For each visited domain, follow rules specified in /robots.txt.  You
       should respect the domain owneâ€™s rules and turn this off only for very
       good reasons.

       Whether enabled or disabled, the robots.txt file is downloaded and
       scanned for sitemaps.  These are lists of pages / files available for
       download that not necessarily are available via recursive scanning.

       This behavior can be switched off by --no-follow-sitemaps.

Recursive Accept/Reject Options
-A acclist, --accept=acclist, -R rejlist, --reject=rejlist
       Specify comma-separated lists of file name suffixes or patterns to
       accept or reject.  Note that if any of the wildcard characters, *,
       ?, [ or ], appear in an element of acclist or rejlist, it is treated as
       a pattern, rather than a suffix.  In this case, you have to enclose the
       pattern into quotes to prevent your shell from expanding it, like in -A
       "*.mp3" or -A '*.mp3'.

--accept-regex=urlregex, --reject-regex=urlregex
       Specify a regular expression to accept or reject file name.

--regex-type=regexype
       Specify a regular expression type.  Possible types are posix or pcre.
       Note that to be able to use pcre type, wget2 has to be compiled with
       libpcre support.

--filter-urls
       Apply the accept and reject filters on the URL before downloading a
       download.

-D domain-list, --domains=domain-list
       Set domains to be followed.  domain-list is a comma-separated list of
       domains.  Note that it does not turn on -H.

--exclude-domains=domain-list
       Specify the domains that are not to be followed.

--follow-sitemaps
       Parsing the sitemaps from robots.txt and follow the links. (default:
       on).

       This option is on for recursive downloads whether you specify --robots
       or --no-robots.  Following the URLs found in sitemaps can be switched
       off with --no-follow-sitemaps.

--follow-tags=list
       Wget2 has an internal table of HTML tag / attribute pairs that it
       considers when looking for linked documents during a recursive
       retrieval.  If a user wants only a subset of those tags to be
       considered, however, he or she should be specify such tags in a comma-
       separated list with this option.

--ignore-tags=list
       This is the opposite of the --follow-tags option.  To skip certain HTML
       tags when recursively looking for documents to download, specify them
       in a comma-separated list.

       In the past, this option was the best bet for downloading a single page
       and its requisites, using a commandline like:

            wget2 --ignore-tags=a,area -H -k -K -r https://

       however, the author of this option came across a page with tags like &lt;LINK&gt;
       and came to the realization that specifying tags to ignore was not
       enough. One can't rely that every page will contain the minimum
       stylesheets needed not be downloaded.  Thus this option for downloading a
       single page and its requisites is the dedicated --page-requisites
       option.

--ignore-case
       Ignore case when matching files and directories.  This influences the
       behavior of -R, -A, -I, and -X options.  For example, with this option,
       -A *.txt.gz will match file1.txt.gz and file2.txt.gz, but not file3.txt.bz.
       ... The quotes in the example are to prevent the shell from expanding
       the pattern.
```

```
    Enable spanning across hosts when doing recursive retrieving.

-L, --relative [Not implemented yet]
    Follow relative links only.  Useful for retrieving a specific home page
    without any distractions, not even those from the same hosts.

-I list, --include-directories=list
    Specify a comma-separated list of directories you wish to follow when
    downloading.  Elements of the list may contain wildcards.

            wget2 -r https://webpage.domain --include-directories=*/pub/*/

    Please keep in mind that */pub/*/ is the same as /*/pub/*/ and that it
    matches directories, not strings.  This means that */pub doesnâ€™t affect
    files contained at e.g. /directory/something/pub but /pub/* matches
    every subdir of /pub.

-X list, --exclude-directories=list
    Specify a comma-separated list of directories you wish to exclude from
    download.  Elements of the list may contain wildcards.

            wget2 -r https://gnu.org --exclude-directories=/software

-I / -X combinations
    Please be aware that the behavior of this combination of flags works
    slightly different than in wget1.x.

    If -I is given first, the default is `exclude all'.  If -X is given
    first, the default is `include all'.

    Multiple -I/-X options are processed `first to last'.  The last match
    is relevant.

            Example: `-I /pub -X /pub/trash` would download all from /pub/ except from /pub/trash.
            Example: `-X /pub -I /pub/important` would download all except from /pub where only /pub/important would be downloaded.

    To reset the list (e.g. to ignore -I/-X from .wget2rc files) use --no-
    include-directories or --no-exclude-directories.

-np, --no-parent
    Do not ever ascend to the parent directory when retrieving recursively.
    This is a useful option, since it guarantees that only the files below
    a certain hierarchy will be downloaded.

--filter-mime-type=list
    Specify a comma-separated list of MIME types that will be downloaded.
    Elements of list may contain wildcards.  If a MIME type starts with the
    character `!' it wonâ€™t be downloaded, this is useful when trying to
    download something with exceptions.  If server doesnâ€™t specify the MIME
    type of a file it will be considered as `application/octet-stream'.
    For example, download everything except images:

            wget2 -r https:/// --filter-mime-type=*,\!image/*

    It is also useful to download files that are compatible with an
    application of your system.  For instance, download every file that is
    compatible with LibreOffice Writer from a website using the recursive
    mode:

            wget2 -r https:/// --filter-mime-type=$(sed -r '/^MimeType=/!d;s/^MimeType=//;s/;/,/g' /usr/share/applications/libreoffice-writer.desktop)

Plugin Options
--list-plugins
    Print a list all available plugins and exit.

--local-plugin=file
    Load file as plugin.

--plugin=name
    Load a plugin with a given name from the configured plugin directories.

--plugin-dirs=directories
    Set plugin directories.  directories is a comma-separated list of
    directories.

--plugin-help
    Print the help messages from all loaded plugins.

--plugin-opt=option
    Set a plugin specific command line option.

    option is in the format .
```