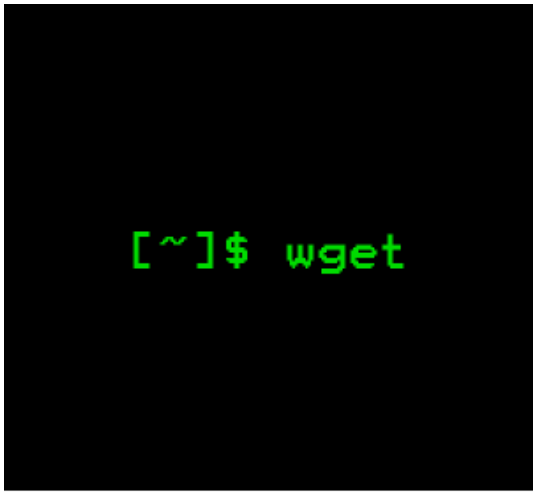
# Linux wget command

Updated: 04/26/2017 by Computer Hope





http://www.computerhope.com

## About wget

**wget** stands for "web get". It is a command-line utility which downloads files over a network.

## Description

**wget** is a free utility for non-interactive download of files from the web. It supports HTTP, HTTPS, and FTP protocols, as well as retrieval through HTTP proxies.

**wget** is non-interactive, meaning that it can work in the background, while the user is not logged on, which allows you to start a retrieval and disconnect from the system, letting **wget** finish the work. By contrast, most web browsersrequire constant user interaction, which make transferring a lot of data difficult.

**wget** can follow links in HTML and XHTML pages and create local versions of remote websites, fully recreating the directory structure of the original site, which is sometimes called "recursive downloading.'' While doing that, **wget** respects the Robot Exclusion Standard (robots.txt). **wget** can be instructed to convert the links in downloaded HTML files to the local files for offline viewing.

**wget** 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 regetting, it will instruct the server to continue the download from where it left off.

## Overview

The simplest way to use wget is to provide it with the location of a file to download over HTTP. For example, to download the file **http://website.com/files/file.zip**, this command:

```
wget http://website.com/files/file.zip
```

...would download the file into the working directory.

There are many options that allow you to use **wget** in different ways, for different purposes. These are outlined below.

## Installing wget

If your operating system is Ubuntu, or another Debian-based Linux distribution which uses APT for package management, you can install **wget** with **apt-get**:

```
sudo apt-get install wget
```

For other operating systems, see your package manager's documentation for information about how to locate the **wget** binary package and install it. Or, you can install it from source from the GNU website at http://www.gnu.org/software/wget/.

## wget syntax

```
wget [option]... [URL]...
```

## Basic Startup Options

| | |
|---|---|
| **-V**, **--version** | Display the version of **wget**, and exit. |
| **-h**, **--help** | Print a help message describing all of **wget**'s command-line options, and exit. |
| **-b**, **--background** | Go to background immediately after startup. If no output file is specified |

| | |
|---|---|
| | via the **-o**, output is redirected to **wget-log**. |
| **-e***command*, **--execute***command* | Execute *command* as if it were a part of the file **.wgetrc**. A command thus invoked will be executed after the commands in **.wgetrc**, thus taking precedence over them. |

## Logging and Input File Options

| | |
|---|---|
| **-o** *logfile*, **--output-file=***logfile* | Log all messages to *logfile*. The messages are normally reported to standard error. |
| **-a** *logfile*, **--append-output=***logfile* | Append to *logfile*. This option 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 **wget** if it does not work properly. Your system administrator may have chosen to compile **wget** without debug support, in which case **-d** will not work.<br><br>Note that compiling with debug support is always safe; **wget** compiled with the debug support will not print any debug info unless requested with **-d**. |
| **-q**, **--quiet** | Turn off **wget**'s output. |
| **-v**, **--verbose** | Turn on verbose output, with all the available data. The default output is |

verbose.

**-nv**, **--non-verbose**

Non-verbose output. Turn off verbose without being completely quiet (use **-q** for that), which means that error messages and basic information still get printed.

**-i** *file*, **--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. If **--force-html** is not specified, then *file* should consist of a series of URLs, one per line.

However, 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 **<base href="***url***">** to the documents or by specifying **--base=***url* on the command line.

If the file is an external one, the document will be automatically treated as HTML if the **Content-Type** is "**text/html**". Furthermore, the file's location will be implicitly used as **base href** if none was specified.

**-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 **<base href="***url***">** to HTML, or using the **--base command-line** option. |
| **-B** *URL*<br>**--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 **--force-html**, or when the input file was fetched remotely from a server describing it as HTML). This option is equivalent to the presence of a "**BASE**" tag in the HTML input file, with URL as the value for the "**href**" attribute.<br><br>For instance, if you specify **http://foo/bar/a.html** for URL, and **wget** reads **../baz/b.html** from the input file, it would be resolved to **http://foo/baz/b.html**. |
| **--config=***FILE* | Specify the location of a startup file you want to use. |

## 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. |
| **-t** *number*, **--tries=***number* | Set number of retries to *number*. Specify **0** or **inf** for infinite retrying. The default is to retry **20** times, with the exception of fatal errors like "connection refused'' or "not found'' (404), which are not retried. |

**-O** *file*, **--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 "**./-**" to print to a file literally named "**-**".)

Use of **-O** is not intended to mean "use the name file instead of the one in the URL;" rather, it is analogous to shell redirection: **wget -O file http://foo** is intended to work like **wget -O - http://foo > file**; *file* will be truncated immediately, and all downloaded content will be written there.

For this reason, **-N** (for timestamp-checking) is not supported in combination with **-O**: since file is always newly created, it will always have a very new timestamp. A warning will be issued if this combination is used.

Similarly, using **-r** or **-p** with **-O** may not work as you expect: **wget** 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. This was disabled in version 1.11, but has been reinstated (with a warning) in 1.11.2, as there are some cases where this behavior can actually have some use.

Note that a combination with **-k** is only permitted when downloading a single document, as in that case it will just convert all relative URIs to external ones; **-k** makes no sense for multiple URIs when they're all being downloaded to a single file; **-k** can be used only when the output is a regular file.

**-nc**, **--no-clobber**

If a file is downloaded more than once in the same directory, **wget**'s behavior depends on a few options, including **-nc**. In certain cases, the local file will be "clobbered" (overwritten), upon repeated download. In other cases it will be preserved.

When running **wget** without **-N**, **-nc**, or **-r**, 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. When **-nc** is specified, this behavior is suppressed, and **wget** will refuse to download newer copies of file. Therefore, "no-clobber" is 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 being turned off.

When running **wget** with **-r**, but without **-N** or **-nc**, re-downloading a file will result in the new copy overwriting the old. 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 **wget** with **-N**, with or without **-r**, 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**.

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.

**-c**, **--continue**

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

```
wget -c ftp://sunsite.doc.ic.ac.uk/ls-lR.Z
```

If there is a file named **ls-lR.Z** in the current directory, **wget** 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 **wget** to retry downloading a file should the connection be lost midway through, which is the default behavior. **-c** only affects resumption of downloads started prior to this invocation of **wget**, and whose local files are still sitting around.

Without **-c**, the previous example would just download the remote file to **ls-lR.Z.1**, leaving the truncated **ls-lR.Z** file alone.

Beginning with **wget** 1.7, if you use **-c** on a non-empty file, and it turns out that the server does not support continued downloading, **wget** 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.

Also, beginning with **wget** 1.7, if you use **-c** on a file that is of equal size as the one on the server, **wget** 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 "continuing" is not meaningful, no download occurs.

On the other hand, while using **-c**, any file that's bigger on the server than locally will be considered an incomplete download and only (length(remote) - length(local)) 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 **wget -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, you'll end up with a garbled file. **wget** 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 "incomplete download" candidate.

Another instance where you'll get a garbled file if you try to use **-c** is if you have a lame HTTP proxy that inserts a "transfer interrupted" string into the local file. In the future a "rollback" option may be added to deal with this case.

Note that **-c** only works with FTP servers and with HTTP servers that support the "Range" header.

**--progress=***type*

Select the progress indicator you want to use. Legal indicators are "**dot**" and "**bar**".

The "**bar**" indicator is used by default. It draws an ASCII progress bar graphics (a.k.a "thermometer" display) indicating the status of retrieval. If the output is not a TTY, the "**dot**" bar will be used by default.

Use **--progress=dot** to switch to the "**dot**" display. It traces the retrieval by printing dots on the screen, each dot representing a fixed amount of downloaded data.

When using the dotted retrieval, you may also set the style by specifying the type as **dot:style**. Different styles assign different meaning to one dot. With the "**default**" style each dot represents 1 K, there are ten dots in a cluster and 50 dots in a line. The "**binary**" style has a more "computer"-like orientation: 8 K dots, 16-dots clusters and 48 dots per line (which makes for 384 K lines). The "**mega**" style is suitable for downloading very large files; each dot represents 6 4K retrieved, there are eight dots in a cluster, and 48 dots on each line (so each line contains 3 M).

Note that you can set the default style using the **progress** command in **.wgetrc**. That setting may be overridden from the command line. The exception is that, when the output is not a TTY, the "**dot**" progress will be

favored over "**bar**". To force the bar output, use **--progress=bar:force**.

**-N**, **--timestamping**

Turn on time stamping. Output file will have timestamp matching remote copy; if file already exists locally, and remote file is not newer, no download will occur.

**--no-use-server-timestamps**

Don't set the local file'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, which allows the use of **--timestamping** on subsequent invocations of **wget**. However, it is sometimes useful to base the local file'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 headers sent by HTTP servers and responses sent by FTP servers.

**--spider**

When invoked with this option, **wget** 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 **wget** to check your bookmarks:

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

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

**-T** *seconds*, **--timeout=***seconds*

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

When interacting with the network, **wget** 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 don'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. Reads that take longer will fail. The default value for read timeout is 900 seconds.

**--limit-rate=***amount*

Limit the download speed to *amount* bytes per second. The *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 20 KB/s. This option is useful when, for whatever reason, you don't want **wget** 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 **wget** 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 don't be

| | |
|---|---|
| | surprised if limiting the rate doesn'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 **m** suffix, in hours using **h** suffix, or in days using **d** suffix.<br><br>Specifying a large value for this option is useful if the network or the destination host is down, so that **wget** 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** (see below). |
| **--waitretry=***seconds* | If you don't want **wget** to wait between every retrieval, but only between retries of failed downloads, you can use this option. **wget** 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. Therefore, a value of **10** will actually make **wget** wait up to (1 + 2 + ... + 10) = 55 seconds per file.<br><br>By default, **wget** will assume a value of **10** seconds. |
| **--random-wait** | Some websites may perform log analysis to identify retrieval programs such as **wget** by looking for statistically significant similarities in the time between requests. This option causes the time between requests to vary between 0 and 2**wait* seconds, where *wait* was specified using the **--wait** option, to mask **wget**'s presence from such analysis. |
| **--no-proxy** | Don't use proxies, even if the appropriate ***_proxy**environment variable is defined. |
| **-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 **wget -Q10k ftp://wuarchive.wustl.edu/ls-lR.gz**, all of the **ls-lR.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 **wget -Q2m -i sites**; download will be aborted when the quota is exceeded.

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

**--no-dns-cache**

Turn off caching of DNS lookups.
Normally, **wget** remembers the addresses it looked up from DNS so it doesn't have to repeatedly contact the DNS server for the same (typically small) set of addresses it retrieves. This cache exists in memory only; a new **wget** 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 **wget**. With this option **wget** issues a new DNS lookup (more precisely, a new call to "**gethostbyname**" or "**getaddrinfo**") 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.

**--restrict-file-names=***modes*

Change which characters found in remote URLs may show up in local file names generated from those URLs. 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.

By default, **wget** escapes the characters that are not valid 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, either because you are downloading to a non-native partition, or because you want to disable escaping of the control characters.

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 mode is set to **unix**, **wget** escapes the character **/** and the control characters in the ranges **0**-**31** and **128**-**159**. This option is the default on Unix-like OSes.

When mode is set to **windows**, **wget** escapes the characters ****, **|**, **/**, **:**, **?**, **"**, *****, **<**, **>**, and the control characters in the ranges **0**-**31** and **128**-**159**. In addition to this, **wget** 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, an 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 filen ames in UTF-8 (some possible byte values used in UTF-8 byte sequences fall in the range of values designated by **wget** as "controls").

| | |
|---|---|
| | 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 mode can be useful when saving file names 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**, **wget** 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**, **wget** will only connect to IPv6 hosts and ignore A records and IPv4 addresses.<br><br>Neither options should be needed normally. By default, an IPv6-aware **wget** will use the address family specified by the host's DNS record. If the DNS responds with both IPv4 and IPv6 addresses, **wget** will try them in sequence until it finds one it can connect to. (Also, see "**--prefer-family**" option described below.)<br><br>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 **wget** 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.<br><br>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 "**IPv4**", the IPv4 address is used first; when the preferred family is "**IPv6**", the IPv6 address is used first; if the specified value is "**none**", the address order returned by DNS is used without change. |

| | |
|---|---|
| | Unlike **-4** and **-6**, this option doesn'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 doesn'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. |
| **--retry-connrefused** | Consider "connection refused" a transient error and try again. Normally **wget** 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* for both FTP and HTTP file retrieval. These parameters can be overridden using the **--ftp-user** and **--ftp-password** options for FTP connections and the **--http-user** and **--http-password** options for HTTP connections. |
| **--ask-password** | Prompt for a password for each connection established. Cannot be specified when **--password** is being used, because they are mutually exclusive. |
| **--no-iri** | Turn off internationalized URI (IRI) support. Use **--iri** to turn it on. IRI support is activated by default.<br><br>You can set the default state of IRI support using the "**iri**" command in **.wgetrc**. That setting may be overridden from the command line. |
| **--local-encoding=***encoding* | Force **wget** to use encoding as the default system encoding. That affects how **wget** converts URLs specified as arguments from locale to UTF-8 for IRI support.<br><br>**wget** use the function "**nl_langinfo()**" and then the "CHARSET" environment variable to get the locale. If it |

| | |
|---|---|
| | fails, ASCII is used.<br><br>You can set the default local encoding using the "**local_encoding**" command in **.wgetrc**. That setting may be overridden from the command line. |
| **--remote-encoding=***encoding* | Force **wget** to use encoding as the default remote server encoding. That affects how **wget** 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.<br><br>For HTTP, remote encoding can be found in HTTP "**Content-Type**" header and in HTML "**Content-Type http-equiv**" meta tag.<br><br>You can set the default encoding using the "**remoteencoding**" command in **.wgetrc**. That setting may be overridden from the command line. |
| **--unlink** | Force **wget** to unlink file instead of clobbering existing file. This option is useful for downloading to the directory with hardlinks. |

## 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 file names 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. **wget -x http://fly.srk.fer.hr/robots.txt** will save the downloaded file to **fly.srk.fer.hr/robots.txt**. |
| **-nH**, **--no-host-directories** | Disable generation of host-prefixed directories. |

By default, invoking **wget** with **-r http://fly.srk.fer.hr/** will create a structure of directories beginning with **fly.srk.fer.hr/**. This option disables such behavior.

**--protocol-directories**

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

**--cut-dirs=***number*

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

Take, for example, the directory at **ftp://ftp.xemacs.org/pub/xemacs/**. If you retrieve it with **-r**, it will be saved locally under **ftp.xemacs.org/pub/xemacs/**. While the **-nH** option can remove the **ftp.xemacs.org/** part, you are still stuck with pub**/xemacs**, which is where **--cut-dirs** comes in handy; it makes **wget** not "see" *number* remote directory components. Here are several examples of how **--cut-dirs** option works:

| | |
|---|---|
| (no options) | **ftp.xemacs.org/pub/xemacs/** |
| **-nH** | **pub/xemacs/** |
| **-nH --cut-dirs=1** | **xemacs/** |
| **-nH --cut-dirs=2** | **.** |
| **--cut-** | **ftp.xemacs.org/xemacs/** |

**dirs=1**

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 **xemacs/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).

## HTTP Options

**-E**, **--html-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 file name. This option is useful, for instance, when you'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 you're downloading CGI-generated materials. An URL like **http://site.com/article.cgi?25** will be saved as **article.cgi?25.html**.

Note that file names changed in this way will be re-downloaded every time you re-mirror a site, because **wget** can't tell that the local **X.html** file corresponds to remote URL **X** (since it doesn't yet know that the URL produces output of type **text/html** or **application/xhtml+xml**).

As of version 1.12, **wget** will also ensure that any downloaded files of type **text/css** end in the suffix **.css**, and the option was renamed from **--html-extension**, to better reflect its new behavior. The old option name is still acceptable, but should now be considered deprecated.

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-passwd=***password*

Specify the username *user* and *password* on an HTTP server. According to the challenge, **wget** will encode them using either the "basic" (insecure) or the "digest" authentication scheme.

Another way to specify username and password is in the URL itself. Either method reveals your password to anyone who bothers to run **ps**. To prevent the passwords from being seen, store them in **.wgetrc** or **.netrc**, and make sure to protect those files from other users with **chmod**. If the passwords are important, do not leave them lying in those files either; edit the files and delete them after **wget** has started the download.

**--no-cache**

Disable server-side cache. In this case, **wget** will send the remote server an appropriate directive (Pragma: **no-cache**) to get the file from the remote service, rather than returning the cached version. This option 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 "**Set-Cookie**" 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 retrieval. *file* is a text file in the format originally used by Netscape'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 **wget** to send the same cookies your browser sends when communicating with the site. To do this use **--load-cookies**; point **wget** 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 text cookie files in different locations:

| | |
|---|---|
| Netscape 4.x | The cookies are in **~/.netscape/cookies.txt**. |
| Mozilla and Netscape 6.x | Mozilla'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**. |
| Internet Explorer | You can produce a cookie file **wget** can use by using the |

| | |
|---|---|
| | File menu, Import and Export, Export Cookies. Tested with Internet Explorer 5 (wow. that's old), but it is not guaranteed to work with earlier versions. |
| other browsers | 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 **wget** expects. |

If you cannot use **--load-cookies**, there might still be an alternative. If your browser supports a "cookie manager", you can use it to view the cookies used when accessing the site you're mirroring. Write down the name and value of the cookie, and manually instruct **wget** to send those cookies, bypassing the "official" cookie support:

```
wget --no-cookies --header "Cookie: <name>=
<value>"
```

| | |
|---|---|
| **--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 "session cookies"), 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 homepage before you can access some pages. With this option, multiple **wget** 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, **wget** marks them with an expiry timestamp of **0**. **wget**'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.

**--ignore-length**

Unfortunately, some HTTP servers (CGI programs, to be more precise) send out bogus "**Content-Length**" headers, which makes **wget** start to bray like a stuck pig, 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, **wget** will ignore the "**Content-Length**" 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.

```
wget --header='Accept-Charset: iso-8859-2'
     --header='Accept-Language: hr'
     http://fly.srk.fer.hr/
```

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

As of **wget** 1.10, this option can be used to override

headers otherwise generated automatically. This example instructs **wget** to connect to localhost, but to specify **foo.bar** in the "**Host**" header:

```
wget --header="Host: foo.bar"
http://localhost/
```

In versions of **wget** prior to 1.10 such use of **--header** caused sending of duplicate headers.

**--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*

Specify the username *user* and *password* for authentication on a proxy server. **wget** will encode them using the "basic" authentication scheme.

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

**--referer=***url*

Include "**Referer: url**" 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 "**User-Agent**" 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 "**User-Agent**"-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 "**User-Agent**" 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 **wget** not to send the "**User-Agent**" 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. **--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 "**key1=value1&key2=value2**", 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 "**key=value**" data (with appropriate percent-coding) just like everything else. **wget** does not currently support "**multipart/form-data**" for transmitting **POST** data; only "**application/x-www-form-urlencoded**". Only one of **--post-data** and **--post-file** should be specified.

Please be aware that **wget** 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** won't work. It's not quite clear how to work around this limitation inherent in HTTP/1.0. Although HTTP/1.1 introduces chunked transfer that doesn't require knowing the request length in advance, a client can't use chunked unless it knows it's talking to an HTTP/1.1 server. And it can't know that until it receives a response, which in turn requires the request to have been completed, which is sort of a chicken-and-egg problem.

Note that if **wget** is redirected after the POST request is completed, it will not send the POST data to the redirected URL. Because URLs that process POST often respond with a redirection to a regular page, which does not desire or accept POST. It is not completely clear that this behavior is optimal; if it doesn't work out, it might be changed in the future.

This example shows how to log to a server using POST and then proceed to download the desired pages, presumably only accessible to authorized users. First, we log in to the server, which can be done only once.

```
wget --save-cookies cookies.txt --post-data
     'user=foo&password=bar'
http://server.com/auth.php
```

And then we grab the page (or pages) we care about:

```
wget --load-cookies cookies.txt
     -p
http://server.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. |
| **--content-disposition** | If this is set, experimental (not fully-functional) support for "**Content-Disposition**" headers is enabled. This option can currently result in extra round-trips to the server for a "**HEAD**" request, and is known to suffer from a few bugs, which is why it is not currently enabled by default.<br><br>This option is useful for some file-downloading CGI programs that use "**Content-Disposition**" headers to describe what the name of a downloaded file should be. |
| **--trust-server-names** | If this is set, 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, **wget** will send Basic HTTP authentication information (plaintext username and password) for all requests, just like **wget** 1.10.2 and prior did by default.<br><br>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. |

## HTTPS (SSL/TLS) Options

To support encrypted HTTP (HTTPS) downloads, **wget** must be compiled with an external SSL library, currently OpenSSL. If **wget** is compiled without SSL support, none of these options are available.

| | |
|---|---|
| **--secure-protocol=***protocol* | Choose the secure |

protocol to be used. Legal values
are **auto**, **SSLv2**, **SSLv3**, and **TLSv1**. If **auto** is used, the SSL library is given the liberty of choosing the appropriate protocol automatically, which is achieved by sending an SSLv2 greeting and announcing support for SSLv3 and TLSv1, which the default.

Specifying SSLv2, SSLv3, or TLSv1 forces the use of the corresponding protocol. This option is useful when talking to old and buggy SSL server implementations that make it hard for OpenSSL to choose the correct protocol version. Fortunately, such servers are quite rare.

**--no-check-certificate** Don't check the server certificate against the available certificate authorities. Also, don't require the URL host name to match the common name presented by the certificate.

As of **wget** 1.10, the default is to verify the server'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 "insecure" mode of operation that turns the certificate verification errors into warnings and allows you to proceed.

If you encounter "certificate verification" errors or ones saying that "common name doesn't match requested host name", 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 site's authenticity, or if you really don'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.

**--certificate=***file* Use the client certificate

| | |
|---|---|
| | stored in *file*. This information is needed for servers that are configured to require certificates from the clients that connect to them. Normally a certificate is not required and this switch is optional. |
| **--certificate-type=***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 option 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 *file* as the file with the bundle of certificate authorities ("CA") to verify the peers. The certificates must be in PEM format.<br><br>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 **--ca-directory** is more efficient than **--ca-certificate** 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.

**--random-file=***file*

Use *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 EGD (see **--egd-file** below) 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 none of those are available, it is likely that SSL encryption will not be usable.

If you're getting the "Could not seed OpenSSL PRNG; disabling SSL." error, you should provide random data using some of the methods described above.

**--egd-file=***file*

Use file as the EGD socket. EGD stands for Entropy Gathering Daemon, a user-space program that collects data from various 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-repeating randomness to seed the random number generator used to produce cryptographically strong keys.

OpenSSL allows the user to specify his own source of entropy using the

"RAND_FILE" 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**.

## FTP Options

**--ftp-user=***user*, **--ftp-password=***password*

Specify the username *user* and *password* on an FTP server. Without this, or the corresponding startup option, the password defaults to **-wget@**, normally used for anonymous FTP.

Another way to specify username and password is in the URL itself. Either method reveals your password to anyone who bothers to run **ps**. To prevent the passwords from being seen, store them in **.wgetrc** or **.netrc**, and make sure to protect those files from other users with **chmod**. If the passwords are important, do not leave them lying in those files either; edit the files and delete them after **wget** has started the download.

**--no-remove-listing**

Don't remove the temporary **.listing** files generated by FTP retrievals. Normally, these files contain the raw directory listings received from FTP servers. Not removing them can be useful for debugging purposes, or when you want to be able to easily check on the contents of remote server directories (e.g. to verify that a mirror you're running is complete).

Note that even though **wget** writes to a known file name for this file, this is not a security hole in the scenario of a user making **.listing** a symbolic link to **/etc/passwd** or something and asking root to run **wget** in his or her directory. Depending on the options used, either **wget** will refuse to write to **.listing**, making the globbing/recursion/time-stamping operation fail, or the symbolic link will be deleted and replaced with the actual .listing file, or the listing will be written to a .listing.number file.

Even though this situation isn't a problem, though, root should never run **wget** in a non-trusted user's directory. A user could do something as simple as linking **index.html** to **/etc/passwd** and asking root to run **wget** with **-N** or **-r** so the file will be overwritten.

**--no-glob**

Turn off FTP globbing. Globbing refers to the use of shell-like special characters (wildcards), like *****, **?**, **[** and **]** to retrieve more than one file from the same directory at once, like:

```
wget
ftp://gnjilux.srk.fer.hr/*.msg
```

By default, globbing will be turned on if the URL contains a globbing character. This option may be used to turn globbing on or off permanently.

You may have to quote the URL to protect it from being expanded by your shell. Globbing makes **wget** look for a directory listing, which is system-specific. This is why it currently works only with Unix FTP servers (and the ones emulating Unix **ls** output).

**--no-passive-ftp**

Disable the use of the passive FTP transfer mode. Passive FTP mandates that the client connect to the server to establish the data connection rather than the other way around.

If the machine is connected to the Internet directly, both passive and active FTP should work equally well. Behind most firewall and NAT configurations passive FTP has a better chance of working. However, in some rare firewall configurations, active FTP actually works when passive FTP doesn't. If you suspect this to be the case, use this option, or set "**passive_ftp=off**" in your init file.

**--retr-symlinks**

Usually, when retrieving FTP directories recursively and a symbolic link is encountered, the linked-to file is not downloaded. Instead, a matching symbolic link is created on the local filesystem. The pointed-to file will not be downloaded unless this recursive

retrieval would have encountered it separately and downloaded it anyway.

When **--retr-symlinks** is specified, however, symbolic links are traversed and the pointed-to files are retrieved. At this time, this option does not cause **wget** to traverse symlinks to directories and recurse through them, but in the future it should be enhanced to do this.

Note that when retrieving a file (not a directory) because it was specified on the command-line, rather than because it was recursed to, this option has no effect. Symbolic links are always traversed in this case.

## Recursive Retrieval Options

**-r**, **--recursive** — Turn on recursive retrieving.

**-l** *depth*, **--level=***depth* — Specify recursion maximum depth level *depth*. The default maximum *depth* is **5**.

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

```
wget -r -nd --delete-after
http://whatever.com/~popular/page/
```

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

Note that **--delete-after** deletes files on

the local machine. It does not issue the **DELE** FTP command to remote FTP sites, for instance. Also, note that when **--delete-after** is specified, **--convert-links** is ignored, so **.orig** files are 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. Note that when **--output-document** is specified, **--convert-links** is ignored. Each link will be changed in one of the two ways:

1. The links to files that have been downloaded by **wget** 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 **wget** 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 **http://hostname/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 links ensures that you can move the downloaded hierarchy to another directory.

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

**-K**, **--backup-converted** When converting a file, backup the original version with an **.orig** suffix. Affects the behavior of **-N**.

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

**-p**, **--page-requisites** This option causes **wget** to download all the files that are necessary to properly display a given HTML page. Including 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 **wget** does not ordinarily distinguish between external and inlined documents, one is generally left with "leaf documents'' that are missing their requisites.

For instance, say document **1.html** contains an **<IMG>** tag referencing **1.gif** and an **<A>** tag 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:

```
wget -r -l 2 http://<site>/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 **wget** is counting the number of hops (up to 2) away from **1.html** to determine where to stop the recursion. However, with this command:

```
wget -r -l 2 -p
http://<site>/1.html
```

all the above files and **3.html**'s requisite **3.gif** will be downloaded. Similarly,

```
wget -r -l 1 -p
http://<site>/1.html
```

will cause **1.html**, **1.gif**, **2.html**, and **2.gif** to be downloaded. One might

think that:

```
wget -r -l 0 -p
http://<site>/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 command-line or in a **-i** URL input file) and its (or their) requisites, leave off **-r** and **-l**:

```
wget -p http://<site>/1.html
```

Note that **wget** 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 lot displays properly locally, this author likes to use a few options in addition to **-p**:

```
wget -E -H -k -K -p
http://<site>/<document>
```

To finish off this topic, it's worth knowing that **wget**'s idea of an external document link is any URL specified in an "**<A>**" tag, an "**<AREA>**" tag, or a "**<LINK>**" tag other than "**<LINK REL="stylesheet">**".

**--strict-comments**

Turn on strict parsing of HTML comments. The default is to terminate comments at the first occurrence of **-->**.

According to specifications, HTML comments are expressed as SGML declarations. Declaration is special markup that begins with **<!** and ends with **>**, such as **<!DOCType ...>**, that may contain comments between a pair of **--** delimiters.
HTML comments are "empty declarations", SGML declarations without any non-comment text. Therefore, **<!--foo-->** is a valid comment, and so is **<!--one-- --two-->**, but **<!--1--2-->** is not.

On the other hand, most HTML writers don't perceive comments as anything other than text delimited with **<!-- and -->**, which is not quite the same. For example, something like **<!------------>** works as a valid comment as long as the number of dashes is a multiple of four. If not, the comment technically lasts until the next **--**, which may be at the other end of the document. Because of this, many popular browsers completely ignore the specification and implement what users have come to expect: comments delimited with **<!-- and -->**.

Until version 1.9, **wget** interpreted comments strictly, which resulted in missing links in many web pages that displayed fine in browsers, but had the misfortune of containing non-compliant comments. Beginning with version 1.9, **wget** has joined the ranks of clients

that implements "naïve'' comments, terminating each comment at the first occurrence of **-->**.

If, for whatever reason, you want strict comment parsing, use this option to turn it on.

## 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 will be treated as a pattern, rather than a suffix. |
| **-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-ftp** | Follow FTP links from HTML documents. Without this option, **wget** will ignore all the FTP links. |
| **--follow-tags=***list* | **wget** 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 option 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 command-line like:

```
wget --ignore-
tags=a,area -H -k -K -r
http://<site>/<document>
```

However, the author of this option came across a page with tags like "**<LINK REL="home" HREF="/">**" and came to the realization that specifying tags to ignore was not enough. One can't just tell **wget** to ignore "**<LINK>**", because then stylesheets will not be downloaded. Now the best bet 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, as well as globbing implemented when downloading from FTP sites. For example, with this option, **-A *.txt** will match **file1.txt**, but also **file2.TXT**, **file3.TxT**, and so on. |
| **-H--span-hosts** | Enable spanning across hosts when doing recursive retrieving. |
| **-L--relative** | Follow relative links only. Useful for retrieving a specific homepage without any distractions, not even those from the same hosts. |
| **-I** *list*, **--include-directories=***list* | Specify a comma-separated list of directories you want to follow when downloading. Elements of *list* may contain wildcards. |
| **-X** *list*, **--exclude-directories=***list* | Specify a comma-separated list of directories you want to exclude from download. Elements of *list* may contain wildcards. |
| **-np**, **--no-parent** | Do not ever ascend to the parent directory when retrieving recursively. This option is a useful option, since it guarantees that only the files below a certain hierarchy will be downloaded. |

## Files

**/etc/wgetrc** Default location of the global startup file.

**.wgetrc** User startup file.

## wget examples

```
wget https://www.computerhope.com/
```

Download the default homepage file (index.htm) from **www.computerhope.com**. The file will be saved to the working directory.

```
wget --limit-rate=200k http://www.example.org/files/archive.zip
```

Download the file **archive.zip** from **www.example.org**, and limit bandwidth usage of the download to **200k/s**.

```
wget -c http://www.example.org/files/archive.zip
```

Download **archive.zip** from **example.org**, and if a partial download exists in the current directory, resume the download where it left off.

```
wget -b http://www.example.org/files/archive.zip
```

Download **archive.zip** in the background, returning you to the command prompt in the interim.

```
wget --spider http://www.example.org/files/archive.zip
```

Uses "web spider" mode to check if a remote file exists. Output will resemble the following:

```
Spider mode enabled. Check if remote file exists.
HTTP request sent, awaiting response... 200 OK
Length: 1206 (1.2K) [application/zip]
Remote file exists.
```

```
wget --mirror -p --convert-links -P ./example-mirror http://www.example.org
```

Download a complete mirror of the website **www.example.org** to the folder **./example-mirror** for local viewing.

```
wget -Q5m http://www.example.org/files/archive.zip
```

Stop downloading **archive.zip** once five megabytes have been successfully transferred. This transfer can then later be resumed using the **-c** option.

## Related commands

**curl** — Transfer data to or from a server.