Introduction to robots.txt

What is a robots.txt file?

A robots.txt file tells search engine crawlers which pages or files the crawler can or can't request from your site. This is used mainly to avoid overloading your site with requests; **it is not a mechanism for keeping a web page out of Google.** To keep a web page out of Google, you should use `noindex` directives (/search/reference/robots_meta_tag), or password-protect your page.

What is robots.txt used for?

robots.txt is used primarily to manage crawler traffic to your site, and *usually* to keep a page off Google, depending on the file type:

Page Type	Traffic management	Hide from Google	Description
Web page	✓	🚫	For web pages (HTML, PDF, or other non-media formats that Google can read (https://support.google.com/webmasters/answer/35287)), robots.txt can be used to manage crawling traffic if you think your server will be overwhelmed by requests from Google's crawler, or to avoid crawling unimportant or similar pages on your site. **You should not use robots.txt as a means to hide your web pages from Google Search results.** This is because, if other pages point to your page with descriptive text, your page could still be indexed without visiting the page. If you want to block your page from search results, use another method such as password protection or a `noindex` (/search/reference/robots_meta_tag) directive. **If your web page is blocked with a robots.txt file**, it can still appear in search results, but the search result will not have a description and look something like this (https://support.google.com/webmasters/answer/7489871). Image files, video files, PDFs, and other non-HTML files will be excluded. If you see this search result for your page and want to fix it, remove the robots.txt entry blocking the page. If you want to hide the page completely from search, use another method (#robotted-but-indexed).

Page Type	Traffic management	Hide from Google	Description
Media file	✓	✓	Use robots.txt to manage crawl traffic, and also to prevent image, video, and audio files from appearing in Google search results. (Note that this won't prevent other pages or users from linking to your image/video/audio file.) • Read more about preventing images from appearing on Google. (/search/docs/advanced/crawling/prevent-images-on-your-page) • Read more about preventing video files from appearing on Google. (/search/docs/advanced/guidelines/video#block_video)
Resource file	✓	✓	You can use robots.txt to block resource files such as unimportant image, script, or style files, **if you think that pages loaded without these resources will not be significantly affected by the loss**. However, if the absence of these resources make the page harder for Google's crawler to understand the page, you should not block them, or else Google won't do a good job of analyzing pages that depend on those resources.

I use a site hosting service

If you use a website hosting service, such as Wix, Drupal, or Blogger, you might not need to (or be able to) edit your robots.txt file directly. Instead, your provider might expose a search settings page or some other mechanism to tell search engines whether or not to crawl your page.

To see if your page has been crawled by Google, search for the page URL in Google.

If you want to hide (or unhide) your page from search engines, add (or remove) any page login requirements that might exist, and search for instructions about modifying your page visibility in search engines on your hosting service, for example: wix hide page from search engines (https://www.google.co.il/search?q=wix+change+robots.txt&oq=wix+hide+page+from+search+results)

Understand the limitations of robots.txt

Before you create or edit robots.txt, you should know the limits of this URL blocking method. At times, you might want to consider other mechanisms to ensure your URLs are not findable on

the web.

- **Robots.txt directives may not be supported by all search engines**
  The instructions in `robots.txt` files cannot enforce crawler behavior to your site, it's up to the crawler to obey them. While Googlebot and other respectable web crawlers obey the instructions in a `robots.txt` file, other crawlers might not. Therefore, if you want to keep information secure from web crawlers, it's better to use other blocking methods, such as password-protecting private files on your server (/search/docs/advanced/crawling/control-what-you-share).
- **Different crawlers interpret syntax differently**
  Although respectable web crawlers follow the directives in a `robots.txt` file, each crawler might interpret the directives differently. You should know the proper syntax for addressing different web crawlers as some might not understand certain instructions.
- **A robotted page can still be indexed if linked to from other sites**
  While Google won't crawl or index the content blocked by `robots.txt`, we might still find and index a disallowed URL if it is linked from other places on the web. As a result, the URL address and, potentially, other publicly available information such as anchor text in links to the page can still appear in Google search results. To properly prevent your URL from appearing in Google Search results, you should password-protect the files on your server (/search/docs/advanced/crawling/control-what-you-share) or use the noindex meta tag or response header (/search/docs/advanced/crawling/block-indexing) (or remove the page entirely).

Combining multiple crawling and indexing directives might cause some directives to counteract other directiv how to configure these directives properly by reading the Combining crawling with indexing / serving directive n (/search/reference/robots_meta_tag#combining) of the Google Developers documentation.

Testing a page for robots.txt blocks

You can test if a page or resource is blocked by a robots.txt rule (https://support.google.com/webmasters/answer/6062598).

To test for noindex directives, use the URL Inspection tool (https://support.google.com/webmasters/answer/9012289).

Last updated 2021-01-25 UTC.

Create a robots.txt file

use a site hosting service, such as Wix or Blogger, you might not need to create or edit a robots.txt file
ch/docs/advanced/robots/intro#site-host).

Getting started

A robots.txt file lives at the root of your site. So, for site www.example.com, the robots.txt file lives at www.example.com/robots.txt. robots.txt is a plain text file that follows the Robots Exclusion Standard (http://en.wikipedia.org/wiki/Robots_exclusion_standard#About_the_standard). A robots.txt file consists of one or more rules. Each rule blocks (or allows) access for a given crawler to a specified file path in that website.

Here is a simple robots.txt file with two rules, explained below:

```
# Group 1
User-agent: Googlebot
Disallow: /nogooglebot/

# Group 2
User-agent: *
Allow: /

Sitemap: http://www.example.com/sitemap.xml
```

Explanation:

1. The user agent named "Googlebot" crawler should not crawl the folder `http://example.com/nogooglebot/` or any subdirectories.
2. All other user agents can access the entire site. (This could have been omitted and the result would be the same, as full access is the assumption.)
3. The site's Sitemap file (http://www.google.com/support/webmasters/bin/answer.py?answer=156184) is located at http://www.example.com/sitemap.xml

We will provide a more detailed example later.

Basic robots.txt guidelines

Here are some basic guidelines for robots.txt files. We recommend that you read the full syntax of robots.txt files (/search/reference/robots_txt)because the robots.txt syntax has some subtle behavior that you should understand.

Format and location

You can use almost any text editor to create a robots.txt file. The text editor should be able to create standard UTF-8 text files. Don't use a word processor; word processors often save files in a proprietary format and can add unexpected characters, such as curly quotes, which can cause problems for crawlers.

e robots.txt Tester tool (https://support.google.com/webmasters/answer/6062598) to write or edit robots.tx
ır site. This tool enables you to test the syntax and behavior against your site.

Format and location rules:

- The file must be named robots.txt
- Your site can have only one robots.txt file.
- The robots.txt file must be located at the **root** of the website host to which it applies. For instance, to control crawling on all URLs below `http://www.example.com/`, the robots.txt file must be located at `http://www.example.com/robots.txt`. It **cannot** be placed in a subdirectory (for example, at `http://example.com/pages/robots.txt`).

If you're unsure about how to access your website root, or need permissions to do so, contact your web hosting service provider. If you can't access your website root, use an alternative blocking method such as meta tags (/search/docs/advanced/crawling/block-indexing).

- A robots.txt file can apply to *subdomains* (for example, `http://website.example.com/robots.txt`) or on non-standard ports (for example, `http://example.com:8181/robots.txt`).
- Comments are any content after a # mark.

Syntax

- robots.txt must be an UTF-8 encoded text file (which includes ASCII). Using other character sets is not possible.
- A robots.txt file consists of one or more **group**.
- Each **group** consists of multiple **rules** or **directives** (instructions), one directive per line.
- A group gives the following information:
  - Who the group applies to (the **user agent**)
  - Which directories or files that agent *can* access, and/or
  - Which directories or files that agent *cannot* access.
- Groups are processed from top to bottom, and a user agent can match only one rule set, which is the first, most-specific rule that matches a given user agent.
- The **default assumption** is that a user agent can crawl any page or directory not blocked by a `Disallow:` rule.
- Rules are **case-sensitive**. For instance, `Disallow: /file.asp` applies to `http://www.example.com/file.asp`, but not `http://www.example.com/FILE.asp`.

The following directives are used in robots.txt files:

- **`User-agent:`** [*Required, one or more per group*] The name of a search engine *robot* (web crawler software) that the rule applies to. This is the first line for any rule. Most Google user agent names are listed in the Web Robots Database (http://www.robotstxt.org/db.html) or in the Google list of user agents (/search/docs/advanced/crawling/overview-google-crawlers)

. Supports the `*` wildcard for a path prefix, suffix, or entire string. Using an asterisk (`*`) as in the example below will match all crawlers **except the various AdsBot crawlers**, which must be named explicitly. (See the list of Google crawler names (/search/docs/advanced/crawling/overview-google-crawlers).) **Examples:**

```
# Example 1: Block only Googlebot
User-agent: Googlebot
Disallow: /

# Example 2: Block Googlebot and Adsbot
User-agent: Googlebot
User-agent: AdsBot-Google
Disallow: /

# Example 3: Block all but AdsBot crawlers
User-agent: *
Disallow: /
```

- **`Disallow:`** [*At least one or more* `Disallow` *or* `Allow` *entries per rule*] A directory or page, relative to the root domain, that should not be crawled by the user agent. If a page, it should be the full page name as shown in the browser; if a directory, it should end in a `/` mark. Supports the `*` wildcard for a path prefix, suffix, or entire string.
- **`Allow:`** [*At least one or more* `Disallow` *or* `Allow` *entries per rule*] A directory or page, relative to the root domain, that should be crawled by the user agent just mentioned. This is used to override a `Disallow` directive to allow crawling of a subdirectory or page in a disallowed directory. If a page, it should be the full page name as shown in the browser; if a directory, it should end in a `/` mark. Supports the `*` wildcard for a path prefix, suffix, or entire string.
- **`Sitemap:`** [*Optional, zero or more per file*] The location of a sitemap for this website. Must be a fully-qualified URL; Google doesn't assume or check http/https/www.non-www alternates. Sitemaps are a good way to indicate which content Google *should* crawl, as opposed to which content it *can* or *cannot* crawl. Learn more about sitemaps. (/search/docs/advanced/sitemaps/overview) **Example:**

```
Sitemap: https://example.com/sitemap.xml
Sitemap: http://www.example.com/sitemap.xml
```

Other rules are ignored.

Another example file

A robots.txt file consists of one or more groups, each beginning with a `User-agent` line that specifies the target of the groups. Here is a file with two group; inline comments explain each group:

```
# Block googlebot from example.com/directory1/... and example.com/directory2/...
# but allow access to directory2/subdirectory1/...
# All other directories on the site are allowed by default.
User-agent: googlebot
Disallow: /directory1/
Disallow: /directory2/
Allow: /directory2/subdirectory1/

# Block the entire site from anothercrawler.
User-agent: anothercrawler
Disallow: /
```

Full robots.txt syntax

You can find the full robots.txt syntax here (/search/reference/robots_txt). Please read the full documentation, as the robots.txt syntax has a few tricky parts that are important to learn.

Useful robots.txt rules

Here are some common useful robots.txt rules:

Rule	Sample
Disallow crawling of the entire website. Keep in mind that in some situations URLs from the website may still be indexed, even if they haven't been crawled. Note: this does not match the various AdsBot crawlers	`User-agent: *` `Disallow: /`

(/search/docs/advanced/crawling/overview-google-crawlers) , which must be named explicitly.	

Disallow crawling of a directory and its contents by following the directory name with a forward slash. Remember that you shouldn't use robots.txt to block access to private content: use proper authentication instead. URLs disallowed by the robots.txt file might still be indexed without being crawled, and the robots.txt file can be viewed by anyone, potentially disclosing the location of your private content.	`User-agent: *`
	`Disallow: /calendar/`
	`Disallow: /junk/`

Allow access to a single crawler	`User-agent: Googlebot-news`
	`Allow: /`
	
	`User-agent: *`
	`Disallow: /`

Allow access to all but a single crawler	`User-agent: Unnecessarybot`
	`Disallow: /`
	
	`User-agent: *`
	`Allow: /`

Disallow crawling of a single webpage by listing the page after the slash:	`User-agent: *` `Disallow: /private_file.html`
Block a specific image from Google Images:	`User-agent: Googlebot-Image` `Disallow: /images/dogs.jpg`
Block all images on your site from Google Images:	`User-agent: Googlebot-Image` `Disallow: /`
Disallow crawling of files of a specific file type (for example, `.gif`):	`User-agent: Googlebot` `Disallow: /*.gif$`

Disallow crawling of entire site, but show AdSense ads on those pages, disallow all web crawlers other than `Mediapartners-Google`. This implementation hides your pages from search results, but the **Mediapartners-Google** web crawler can still analyze them to decide what ads to show visitors to your site.	`User-agent: *`
	`Disallow: /`
	
	`User-agent: Mediapartners-Google`
	`Allow: /`

Match URLs that end with a specific string, use $. For instance, the sample code blocks any URLs that end	`User-agent: Googlebot` `Disallow: /*.xls$`

with `.xls`:

Last updated 2021-01-15 UTC.

Submit your updated robots.txt to Google

The **Submit** function of the **robots.txt Tester** (https://www.google.com/webmasters/tools/robots-testing-tool) tool allows you to easily update and ask Google to more quickly crawl and use a new robots.txt file for your site. Update and notify Google of changes to your robots.txt file by following the steps below.

1. Click **Submit** in the bottom-right corner of the robots.txt editor. This action opens up a Submit dialog.
2. Download your edited robots.txt code from the **robots.txt Tester** page by clicking **Download** in the **Submit** dialog.
3. Upload your new robots.txt file to the root of your domain as a text file named robots.txt (the URL for your robots.txt file should be `/robots.txt`).

★ **If you do not have permission to upload files to the root of your domain, you should contact your domain manager to make changes.**

For example, if your site home page resides under **`subdomain.example.com/site/example/`**, you likely cannot update the robots.txt file at **`subdomain.example.com/robots.txt`**. In this case, you should contact the owner of **`example.com/`** to make any necessary changes to the robots.txt file.

4. Click **View uploaded version** to see that your live robots.txt is the version that you want Google to crawl.
5. Click **Submit** to notify Google that changes have been made to your robots.txt file and request that Google crawl it.
6. Check that your newest version was successfully crawled by Google by refreshing the page in your browser to update the tool's editor and see your live robots.txt code. After you refresh the page, you can also click the dropdown above the text editor to view the timestamp of when Google first saw the **latest version** of your robots.txt file.

Last updated 2021-01-11 UTC.

Robots FAQs

ve missed an FAQ? Feel free to post in our Google Search Central Help Community ://support.google.com/webmasters/community/) for more help!

General robots questions

Does my website need a robots.txt file?

No. When Googlebot visits a website, we first ask for permission to crawl by attempting to retrieve the robots.txt file. A website without a robots.txt file, robots meta tags or X-Robots-Tag HTTP headers will generally be crawled and indexed normally.

Which method should I use?

It depends. In short, there are good reasons to use each of these methods:

1. robots.txt: Use it if crawling of your content is causing issues on your server. For example, you may want to disallow crawling of infinite calendar scripts. You should not use the robots.txt to block private content (use server-side authentication instead), or handle canonicalization (/search/docs/advanced/robots/search/docs/advanced/guidelines/duplicate-content). If you must be certain that a URL is not indexed, use the robots meta tag or `X-Robots-Tag` HTTP header instead.
2. robots meta tag: Use it if you need to control how an individual HTML page is shown in search results (or to make sure that it's not shown).
3. `X-Robots-Tag` HTTP header: Use it if you need to control how non-HTML content is shown in search results (or to make sure that it's not shown).

Can I use these methods to remove someone else's site?

No. These methods are only valid for sites where you can modify the code or add files. If you want to remove content from a third-party site, you need to contact the website owner to have

them remove the content.

How can I slow down Google's crawling of my website?

You can generally adjust the crawl rate setting (https://support.google.com/webmasters/answer/48620) in your Google Search Console (https://search.google.com/search-console) account.

Robots.txt questions

I use the same robots.txt for multiple websites. Can I use a full URL instead o relative path?

No. The directives in the robots.txt file (with exception of `Sitemap:`) are only valid for relative paths.

Can I place the robots.txt file in a subdirectory?

No. The file must be placed in the topmost directory of the website.

I want to block a private folder. Can I prevent other people from reading my robots.txt file?

No. The robots.txt file may be read by various users. If folders or filenames of content should not be public, they should not be listed in the robots.txt file. It is not recommended to serve different robots.txt files based on the user agent or other attributes.

Do I have to include an `allow` directive to allow crawling?

No, you do not need to include an `allow` directive. The `allow` directive is used to override `disallow` directives in the same robots.txt file.

What happens if I have a mistake in my robots.txt file or use an unsupported directive?

Web-crawlers are generally very flexible and typically will not be swayed by minor mistakes in the robots.txt file. In general, the worst that can happen is that incorrect / unsupported directives will be ignored. Bear in mind though that Google can't read minds when interpreting a robots.txt file; we have to interpret the robots.txt file we fetched. That said, if you are aware of problems in your robots.txt file, they're usually easy to fix.

What program should I use to create a robots.txt file?

You can use anything that creates a valid text file. Common programs used to create robots.txt files are Notepad, TextEdit, vi, or emacs. Read more about creating robots.txt files (/search/docs/advanced/robots/intro). After creating your file, validate it using the robots.txt tester (https://www.google.com/webmasters/tools/robots-testing-tool).

If I block Google from crawling a page using a robots.txt `disallow` directive will it disappear from search results?

Blocking Google from crawling a page is likely to remove the page from Google's index.

However, the `Disallow` directive does not guarantee that a page will not appear in results: Google may still decide, based on external information such as incoming links, that it is relevant. If you wish to explicitly block a page from being indexed, you should instead use the `noindex` robots meta tag or `X-Robots-Tag` HTTP header. In this case, you should not disallow the page in robots.txt, because the page must be crawled in order for the tag to be seen and obeyed.

How long will it take for changes in my robots.txt file to affect my search results?

First, the cache of the robots.txt file must be refreshed (we generally cache the contents for up to one day). Even after finding the change, crawling and indexing is a complicated process that can sometimes take quite some time for individual URLs, so it's impossible to give an exact timeline. Also, keep in mind that even if your robots.txt file is disallowing access to a URL, that URL may remain visible in search results despite that fact that we can't crawl it. If you wish to expedite removal of the pages you've blocked from Google, please submit a removal request via Google Search Console (https://search.google.com/search-console).

How can I temporarily suspend all crawling of my website?

You can temporarily suspend all crawling by returning a HTTP result code of 503 for all URLs, including the robots.txt file. The robots.txt file will be retried periodically until it can be accessed again. We do not recommend changing your robots.txt file to disallow crawling.

My server is not case-sensitive. How can I disallow crawling of some folders completely?

Directives in the robots.txt file are case-sensitive. In this case, it is recommended to make sure that only one version of the URL is indexed using canonicalization methods (/search/docs/advanced/crawling/consolidate-duplicate-urls). Doing this allows you to simplify your robots.txt file. Should this not be possible, we recommended that you list the common combinations of the folder name, or to shorten it as much as possible, using only the first few characters instead of the full name. For instance, instead of listing all upper and lower-case permutations of `/MyPrivateFolder`, you could list the permutations of "/MyP" (if you are certain that no other, crawlable URLs exist with those first characters). Alternately, it may make sense to use a robots meta tag or `X-Robots-Tag` HTTP header instead, if crawling is not an issue.

I return `403 Forbidden` for all URLs, including the robots.txt file. Why is the s still being crawled?

The HTTP result code 403—as all other 4xx HTTP result codes—is seen as a sign that the robots.txt file does not exist. Because of this, crawlers will generally assume that they can crawl all URLs of the website. In order to block crawling of the website, the robots.txt must be returned normally (with a 200 "OK" HTTP result code) with an appropriate `disallow` directive in it.

Robots meta tag questions

Is the robots meta tag a replacement for the robots.txt file?

No. The robots.txt file controls which pages are accessed. The robots meta tag controls whether a page is indexed, but to see this tag the page needs to be crawled. If crawling a page is problematic (for example, if the page causes a high load on the server), you should use the

robots.txt file. If it is only a matter of whether or not a page is shown in search results, you can use the robots meta tag.

Can the robots meta tag be used to block a part of a page from being index

No, the robots meta tag is a page-level setting.

Can I use the robots meta tag outside of a `<head>` section?

No, the robots meta tag currently needs to be in the `<head>` section of a page.

Does the robots meta tag disallow crawling?

No. Even if the robots meta tag currently says `noindex`, we'll need to recrawl that URL occasionally to check if the meta tag has changed.

How does the `nofollow` robots meta tag compare to the `rel="nofollow"` l attribute?

The `nofollow` robots meta tag applies to all links on a page. The `rel="nofollow"` link attribute only applies to specific links on a page. For more information on the `rel="nofollow"` link attribute, please see our Help Center articles on user-generated spam (/search/docs/advanced/guidelines/prevent-comment-spam) and the rel="nofollow" (/search/docs/advanced/guidelines/qualify-outbound-links).

X-Robots-Tag HTTP header questions

How can I check the X-Robots-Tag for a URL?

A simple way to view the server headers is to use a web-based server header checker (https://www.google.com/search?q=server+header+checker) or to use the Fetch as Googlebot (https://support.google.com/webmasters/answer/158587) feature in Google Search Console (https://search.google.com/search-console).

Last updated 2021-01-15 UTC.

Robots.txt Specifications

Abstract

This document details how Google handles the robots.txt file that allows you to control how Google's website crawlers crawl and index publicly accessible websites.

What changed

On July 1, 2019, Google announced (/search/blog/2019/07/rep-id) that the robots.txt protocol is working towards becoming an Internet standard (https://tools.ietf.org/html/draft-koster-rep-00). Those changes are reflected in this document.

List of changes

Here's what changed:

- Removed the "Requirements Language" section in this document because the language is Internet draft specific.
- Robots.txt now accepts all URI-based (https://en.wikipedia.org/wiki/Uniform_Resource_Identifier) protocols.
- Google follows at least five redirect hops. Since there were no rules fetched yet, the redirects are followed for at least five hops and if no robots.txt is found, Google treats it as a 404 for the robots.txt. Handling of logical redirects for the robots.txt file based on HTML content that returns 2xx (frames, JavaScript, or meta refresh-type redirects) is discouraged and the content of the first page is used for finding applicable rules.
- For 5xx, if the robots.txt is unreachable for more than 30 days, the last cached copy of the robots.txt is used, or if unavailable, Google assumes that there are no crawl restrictions.
- Google treats unsuccessful requests or incomplete data as a server error.
- "Records" are now called "lines" or "rules", as appropriate.

- Google doesn't support the handling of `<field>` elements with simple errors or typos (for example, "useragent" instead of "user-agent").
- Google currently enforces a size limit of 500 kibibytes (https://en.wikipedia.org/wiki/Kibibyte) (KiB), and ignores content after that limit.
- Updated formal syntax to be valid Augmented Backus-Naur Form (ABNF) per RFC5234 (https://tools.ietf.org/html/rfc5234) and to cover for UTF-8 characters in the robots.txt.
- Updated the definition of "groups" to make it shorter and more to the point. Added an example for an empty group.
- Removed references to the deprecated Ajax Crawling Scheme.

Basic definitions

Definitions	
Crawler	A crawler is a service or agent that crawls websites. Generally speaking, a crawler automatically and recursively accesses known URLs of a host that exposes content which can be accessed with standard web-browsers. As new URLs are found (through various means, such as from links on existing, crawled pages or from Sitemap files), these are also crawled in the same way.
User-agent	A means of identifying a specific crawler or set of crawlers.
Directives	The list of applicable guidelines for a crawler or group of crawlers set forth in the robots.txt file.
URL	Uniform Resource Locators as defined in RFC 1738 (http://www.ietf.org/rfc/rfc1738.txt).
Google-specific	These elements are specific to Google's implementation of robots.txt and may not be relevant for other parties.

Applicability

The guidelines set forth in this document are followed by all automated crawlers at Google. When an agent accesses URLs on behalf of a user (for example, for translation, manually

subscribed feeds, malware analysis), these guidelines do not need to apply.

File location and range of validity

The robots.txt file must be in the top-level directory of the host, accessible through the appropriate protocol and port number. Generally accepted protocols for robots.txt are all URI-based (https://en.wikipedia.org/wiki/Uniform_Resource_Identifier), and for Google Search specifically (for example, crawling of websites) are "http" and "https". On http and https, the robots.txt file is fetched using a HTTP non-conditional GET request.

Google-specific: Google also accepts and follows robots.txt files for FTP sites. FTP-based robots.txt files are accessed via the FTP protocol, using an anonymous login.

The directives listed in the robots.txt file apply only to the host, protocol and port number where the file is hosted.

RL for the robots.txt file is - like other URLs - case-sensitive.

Examples of valid robots.txt URLs

Robots.txt URL examples

Robots.txt URL examples	
`http://example.com/robots.txt`	**Valid for:** • `http://example.com/` • `http://example.com/folder/file` **Not valid for:** • `http://other.example.com/` • `https://example.com/` • `http://example.com:8181/` This is the general case. It is not valid for other subdomains, protocols or port numbers. It is valid for all files in all subdirectories on the same host, protocol and port number.
`http://www.example.com/robots.txt`	**Valid for:** `http://www.example.com/` **Not valid for:** • `http://example.com/` • `http://shop.www.example.com/` • `http://www.shop.example.com/` A robots.txt on a subdomain is only valid for that subdomain.
`http://example.com/folder/robots.txt`	Not a valid robots.txt file. Crawlers don't check for robots.txt files in subdirectories.
`http://www.müller.eu/robots.txt`	**Valid for:** • `http://www.müller.eu/` • `http://www.xn--mller-kva.eu/` **Not valid for:** `http://www.muller.eu/` IDNs are equivalent to their punycode versions. See also RFC 3492 (http://www.ietf.org/rfc/rfc3492.txt).

Robots.txt URL examples	
`ftp://example.com/robots.txt`	👍 **Valid for:** `ftp://example.com/` 👎 **Not valid for:** `http://example.com/` Google-specific: We use the robots.txt for FTP resources.
`http://212.96.82.21/robots.txt`	👍 **Valid for:** `http://212.96.82.21/` 👎 **Not valid for:** `http://example.com/` (even if hosted on 212.96.82.21) 💡 A robots.txt with an IP-address as the host name is only valid for crawling of that IP-address as host name. It isn't automatically valid for all websites hosted on that IP-address (though it is possible that the robots.txt file is shared, in which case it would also be available under the shared host name).
`http://example.com:80/robots.txt`	👍 **Valid for:** • `http://example.com:80/` • `http://example.com/` 👎 **Not valid for:** `http://example.com:81/` 💡 Standard port numbers (80 for http, 443 for https, 21 for ftp) are equivalent to their default host names. See also **`[portnumbers]`**.
`http://example.com:8181/robots.txt`	👍 **Valid for:** `http://example.com:8181/` 👎 **Not valid for:** `http://example.com/` 💡 Robots.txt files on non-standard port numbers are only valid for content made available through those port numbers.

Handling HTTP result codes

There are generally three different outcomes when robots.txt files are fetched:

- full allow: All content may be crawled.
- full disallow: No content may be crawled.
- conditional allow: The directives in the robots.txt determine the ability to crawl certain content.

Handling HTTP result codes	
2xx (successful)	HTTP result codes that signal success result in a "conditional allow" of crawling.
3xx (redirection)	Google follows at least five redirect hops as defined by RFC 1945 (http://www.ietf.org/rfc/rfc1945.txt) for HTTP/1.0 and then stops and treats it as a 404. Handling of robots.txt redirects to disallowed URLs is discouraged; since there were no rules fetched yet, the redirects are followed for at least five hops and if no robots.txt is found, Google treats it as a 404 for the robots.txt. Handling of logical redirects for the robots.txt file based on HTML content that returns 2xx (frames, JavaScript, or meta refresh-type redirects) is discouraged and the content of the first page is used for finding applicable rules.
4xx (client errors)	All 4xx errors are treated the same way and it's assumed that no valid robots.txt file exists. It is assumed that there are no restrictions. This is a "full allow" for crawling.
★	This includes 401 "Unauthorized" and 403 "Forbidden" HTTP result codes.
5xx (server error)	Server errors are seen as temporary errors that result in a "full disallow" of crawling. The request is retried until a non-server-error HTTP result code is obtained. A 503 (Service Unavailable) error results in fairly frequent retrying. If the robots.txt is unreachable for more than 30 days, the last cached copy of the robots.txt is used. If unavailable, Google assumes that there are no crawl restrictions. To temporarily suspend crawling, it is recommended to serve a 503 HTTP result code. Google-specific: If we are able to determine that a site is incorrectly configured to return 5xx instead of a 404 for missing pages, we treat a 5xx error from that site as a 404.

Handling HTTP result codes	
Unsuccessful requests or incomplete data	Handling of a robots.txt file which cannot be fetched due to DNS or networking issues, such as timeouts, invalid responses, reset or hung up connections, and HTTP chunking errors, is treated as a server error (#server-error).
Caching	robots.txt content is generally cached for up to 24 hours, but may be cached longer in situations where refreshing the cached version is not possible (for example, due to timeouts or 5xx errors). The cached response may be shared by different crawlers. Google may increase or decrease the cache lifetime based on max-age Cache-Control (http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.9.3) HTTP headers.

File format

The expected file format is plain text encoded in UTF-8 (http://en.wikipedia.org/wiki/UTF-8). The file consists of lines separated by CR, CR/LF, or LF.

Only valid lines are considered; all other content is ignored. For example, if the resulting document is an HTML page, only valid text lines are taken into account, the rest are discarded without warning or error.

If a character encoding is used that results in characters being used which are not a subset of UTF-8, this may result in the contents of the file being parsed incorrectly.

An optional Unicode BOM (http://en.wikipedia.org/wiki/Byte_order_mark) (byte order mark) at the beginning of the robots.txt file is ignored.

Each valid line consists of a field, a colon, and a value. Spaces are optional (but recommended to improve readability). Comments can be included at any location in the file using the "#" character; all content after the start of a comment until the end of the line is treated as a comment and ignored. The general format is `<field>:<value><#optional-comment>`. Whitespace at the beginning and at the end of the line is ignored.

The `<field>` element is case-insensitive. The <value> element may be case-sensitive, depending on the <field> element.

Handling of `<field>` elements with simple errors or typos (for example, "useragent" instead of "user-agent") is not supported.

A maximum file size may be enforced per crawler. Content which is after the maximum file size is ignored. Google currently enforces a size limit of 500 kibibytes (https://en.wikipedia.org/wiki/Kibibyte) (KiB). To reduce the size of the robots.txt file, consolidate directives that would result in an oversized robots.txt file. For example, place excluded material in a separate directory.

Formal syntax / definition

Here is an Augmented Backus-Naur Form (ABNF) description, as described in RFC 5234 (https://www.ietf.org/rfc/rfc5234.txt)

```
robotstxt = *(group / emptyline)
group = startgroupline                    ; We start with a user-agent
        *(startgroupline / emptyline)     ; ... and possibly more user-agents
        *(rule / emptyline)               ; followed by rules relevant for UAs


startgroupline = *WS "user-agent" *WS ":" *WS product-token EOL

rule = *WS ("allow" / "disallow") *WS ":" *WS (path-pattern / empty-pattern) EOL

; parser implementors: add additional lines you need (for example, Sitemaps), an
; be lenient when reading lines that don't conform. Apply Postel's law.

product-token = identifier / "*"
path-pattern = "/" *(UTF8-char-noctl)    ; valid URI path pattern; see 3.2.2
empty-pattern = *WS

identifier = 1*(%x2d / %x41-5a / %x5f / %x61-7a)
comment = "#" *(UTF8-char-noctl / WS / "#")
emptyline = EOL
EOL = *WS [comment] NL         ; end-of-line may have optional trailing comment
NL = %x0D / %x0A / %x0D.0A
WS = %x20 / %x09

; UTF8 derived from RFC3629, but excluding control characters
UTF8-char-noctl = UTF8-1-noctl / UTF8-2 / UTF8-3 / UTF8-4
```

```
UTF8-1-noctl    = %x21 / %x22 / %x24-7F  ; excluding control, space, '#'
UTF8-2          = %xC2-DF UTF8-tail
UTF8-3          = %xE0 %xA0-BF UTF8-tail / %xE1-EC 2( UTF8-tail ) /
                  %xED %x80-9F UTF8-tail / %xEE-EF 2( UTF8-tail )
UTF8-4          = %xF0 %x90-BF 2( UTF8-tail ) / %xF1-F3 3( UTF8-tail ) /
                  %xF4 %x80-8F 2( UTF8-tail )
UTF8-tail       = %x80-BF
```

Grouping of lines and rules

One or more `user-agent` lines that is followed by one or more rules. The group is terminated by a `user-agent` line or end of file. The last group may have no rules, which means it implicitly allows everything.

Example groups:

```
user-agent: a
disallow: /c

user-agent: b
disallow: /d

user-agent: e
user-agent: f
disallow: /g

user-agent: h
```

There are four distinct groups specified:

- One group for "a"
- One group for "b"
- One group for both "e" and "f"
- One group for "h"

Except for the last group (group "h"), each group has its own group-member line. The last group (group "h") is empty. Note the optional use of white-space and empty lines to improve readability.

Order of precedence for user agents

Only one group is valid for a particular crawler. The crawler must determine the correct group of lines by finding the group with the most specific user agent that still matches. All other groups are ignored by the crawler. The user agent is case-sensitive. All non-matching text is ignored (for example, both `googlebot/1.2` and `googlebot*` are equivalent to `googlebot`). The order of the groups within the robots.txt file is irrelevant.

If there's more than one group declared for a specific user agent, all the rules from the groups applicable to the specific user agent are combined into a single group.

Examples

Example 1

Assuming the following robots.txt file:

```
user-agent: googlebot-news
(group 1)

user-agent: *
(group 2)

user-agent: googlebot
(group 3)
```

This is how the crawlers would choose the relevant group:

Group followed per crawler	
Googlebot News	The group followed is group 1. Only the most specific group is followed, all others are ignored.

Group followed per crawler	
Googlebot (web)	The group followed is group 3.
Googlebot Images	The group followed is group 3. There is no specific `googlebot-images` group, so the more generic group is followed.
Googlebot News (when crawling images)	>The group followed is group 1. These images are crawled for and by Googlebot News, therefore only the Googlebot News group is followed.
Otherbot (web)	The group followed is group 2.
Otherbot (News)	The group followed is group 2. Even if there is an entry for a related crawler, it is only valid if it is specifically matching.

Example 2

Assuming the following robots.txt file:

```
user-agent: googlebot-news
disallow: /fish

user-agent: *
disallow: /carrots

user-agent: googlebot-news
disallow: /shrimp
```

This is how the crawlers would merge groups relevant to a specific user agent:

```
user-agent: googlebot-news
disallow: /fish
disallow: /shrimp

user-agent: *
disallow: /carrots
```

Also see Google's crawlers and user-agent strings (/search/docs/advanced/crawling/overview-google-crawlers).

Group-member rules

Only standard group-member rules are covered in this section. These rules are also called "directives" for the crawlers. These directives are specified in the form of `directive: [path]` where `[path]` is optional. By default, there are no restrictions for crawling for the designated crawlers. Directives without a `[path]` are ignored.

The `[path]` value, if specified, is to be seen relative from the root of the website for which the robots.txt file was fetched (using the same protocol, port number, host and domain names). The path value must start with "/" to designate the root. The path is case-sensitive. More information can be found in the section "URL matching based on path values" below.

disallow

The `disallow` directive specifies paths that must not be accessed by the designated crawlers. When no path is specified, the directive is ignored.

Usage:

```
disallow: [path]
```

allow

The `allow` directive specifies paths that may be accessed by the designated crawlers. When no path is specified, the directive is ignored.

Usage:

```
allow: [path]
```

URL matching based on path values

The path value is used as a basis to determine whether or not a rule applies to a specific URL on a site. With the exception of wildcards, the path is used to match the beginning of a URL (and any valid URLs that start with the same path). Non-7-bit ASCII characters in a path may be included as UTF-8 characters or as percent-escaped UTF-8 encoded characters per RFC 3986 (http://www.ietf.org/rfc/rfc3986.txt).

Google, Bing, and other major search engines support a limited form of "wildcards" for path values. These are:

- `*` designates 0 or more instances of any valid character.
- `$` designates the end of the URL.

Example path matches	
`/`	Matches the root and any lower level URL
`/*`	Equivalent to `/`. The trailing wildcard is ignored.
`/fish`	**Matches:** • `/fish` • `/fish.html` • `/fish/salmon.html` • `/fishheads` • `/fishheads/yummy.html` • `/fish.php?id=anything` **Does not match:** • `/Fish.asp` • `/catfish` • `/?id=fish` ★ Note the case-sensitive matching.

Example path matches	
`/fish*`	Equivalent to `/fish`. The trailing wildcard is ignored. 👍 **Matches:** • `/fish` • `/fish.html` • `/fish/salmon.html` • `/fishheads` • `/fishheads/yummy.html` • `/fish.php?id=anything` 👎 **Does not match:** • `/Fish.asp` • `/catfish` • `/?id=fish`
`/fish/`	The trailing slash means this matches anything in this folder. 👍 **Matches:** • `/fish/` • `/fish/?id=anything` • `/fish/salmon.htm` 👎 **Does not match:** • `/fish` • `/fish.html` • `/Fish/Salmon.asp`

Example path matches	
`/*.php`	**Matches:** • `/filename.php` • `/folder/filename.php` • `/folder/filename.php?parameters` • `/folder/any.php.file.html` • `/filename.php/` **Does not match:** • `/` (even if it maps to /index.php) • `/windows.PHP`
`/*.php$`	**Matches:** • `/filename.php` • `/folder/filename.php` **Does not match:** • `/filename.php?parameters` • `/filename.php/` • `/filename.php5` • `/windows.PHP`
`/fish*.php`	**Matches:** • `/fish.php` • `/fishheads/catfish.php?parameters` **Does not match:** `/Fish.PHP`

Google-supported non-group-member lines

Google, Bing,and other major search engines support `sitemap`, as defined by sitemaps.org (http://sitemaps.org).

Usage:

```
sitemap: [absoluteURL]
```

`[absoluteURL]` points to a Sitemap, Sitemap Index file, or equivalent URL. The URL does not have to be on the same host as the robots.txt file. Multiple `sitemap` entries may exist. As non-group-member lines, these are not tied to any specific user agents and may be followed by all crawlers, provided it is not disallowed.

Order of precedence for group-member lines

At a group-member level, in particular for `allow` and `disallow` directives, the most specific rule based on the length of the `[path]` entry trumps the less specific (shorter) rule. In case of conflicting rules, including those with wildcards, the least restrictive rule is used.

Sample situations	

`http://example.com/page`	`allow`: `/p`
	`disallow`: `/`
	Verdict: `allow`

`http://example.com/folder/page`	`allow`: `/folder`
	`disallow`: `/folder`
	Verdict: `allow`

`http://example.com/page.htm`	`allow`: `/page`
	`disallow`: `/*.htm`
	Verdict: `undefined`

Sample situations	

`http://example.com/`	`allow`: `/$`
	`disallow`: `/`
	Verdict: `allow`

`http://example.com/page.htm`	`allow`: `/$`
	`disallow`: `/`
	Verdict: `disallow`

Testing robots.txt markup

Google offers two options for testing robots.txt markup:

1. The robots.txt Tester (https://support.google.com/webmasters/answer/6062598) in Search Console.
2. Google's open source robots.txt library (https://github.com/google/robotstxt), which is also used in Google Search.

Last updated 2021-01-15 UTC.