What is a web server?

In this article, we explain what web servers are, how web servers work, and why they are important.

Prerequisites:	You should already know how the Internet works, and understand the difference between a web page, a web site, a web server, and a search engine.
Objective:	You will learn what a web server is and gain a general understanding of how it works.

Summary

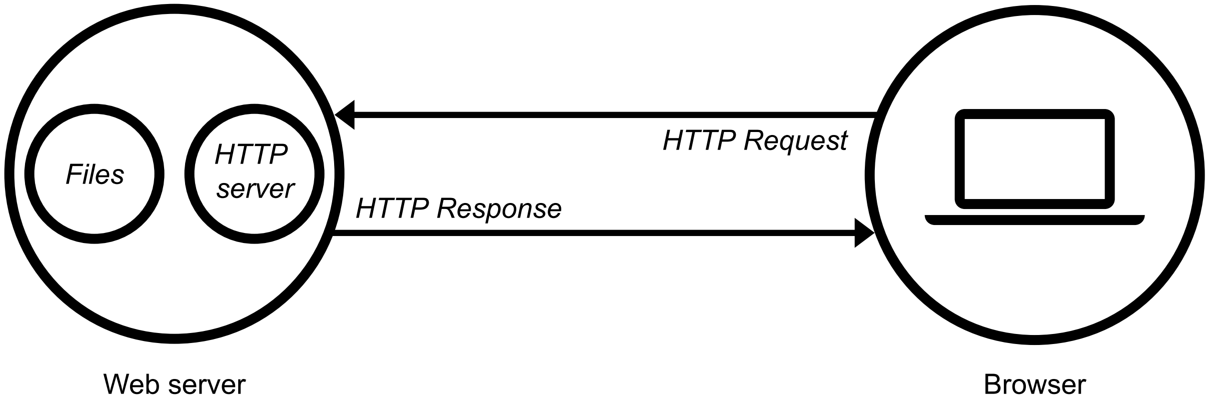
The term *web server* can refer to hardware or software, or both of them working together.

1. On the hardware side, a web server is a computer that stores web server software and a website's component files. (for example, HTML documents, images, CSS stylesheets,

and JavaScript files) A web server connects to the Internet and supports physical data interchange with other devices connected to the web.

2. On the software side, a web server includes several parts that control how web users access hosted files. At a minimum, this is an *HTTP server*. An HTTP server is software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). An HTTP server can be accessed through the domain names of the websites it stores, and it delivers the content of these hosted websites to the end user's device.

At the most basic level, whenever a browser needs a file that is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct (hardware) web server, the (software) *HTTP server* accepts the request, finds the requested document, and sends it back to the browser, also through HTTP. (If the server doesn't find the requested document, it returns a 404 response instead.)



To publish a website, you need either a static or a dynamic web server.

A **static web server**, or stack, consists of a computer (hardware) with an HTTP server (software). We call it "static" because the server sends its hosted files as-is to your browser.

A **dynamic web server** consists of a static web server plus extra software, most commonly an *application server* and a *database*. We call it "dynamic" because the application server updates the hosted files before sending content to your browser via the HTTP server.

For example, to produce the final webpages you see in the browser, the application server might fill an HTML template with content from a database. Sites like MDN or Wikipedia have thousands of webpages. Typically, these kinds of sites are composed of only a few HTML templates and a giant database, rather than thousands of static HTML documents. This setup makes it easier to maintain and deliver the content.

Active learning

There is no active learning available yet. Please, consider contributing.

Deeper dive

To review: to fetch a webpage, your browser sends a request to the web server, which searches for the requested file in its own storage space. Upon finding the file, the server reads it, processes it as-needed, and sends it to the browser. Let's look at those steps in more detail.

Hosting files

First, a web server has to store the website's files, namely all HTML documents and their related assets, including images, CSS stylesheets, JavaScript files, fonts, and video.

Technically, you could host all those files on your own computer, but it's far more convenient to store files all on a dedicated web server because:

- A dedicated web server is typically more available. (up and running)
- Excusing downtime and systems troubles, a dedicated web server is always connected to the Internet.
- A dedicated web server can have the same IP address all the time. This is known as a *dedicated IP address*. (not all ISPs provide a fixed IP address for home lines)
- A dedicated web server is typically maintained by a third-party.

For all these reasons, finding a good hosting provider is a key part of building your website. Examine the various services companies offer. Choose one that fits your needs and budget. (Services range from free to thousands of dollars per month.) You can find more details in this article.

Once you have web hosting service, you must upload your files to your web server.

Communicating through HTTP

Second, a web server provides support for HTTP (**H**yper**t**ext **T**ransfer **P**rotocol). As its name implies, HTTP specifies how to transfer hypertext (linked web documents) between two computers.

A Protocol is a set of rules for communication between two computers. HTTP is a textual, stateless protocol.

Textual

All commands are plain-text and human-readable.

Stateless

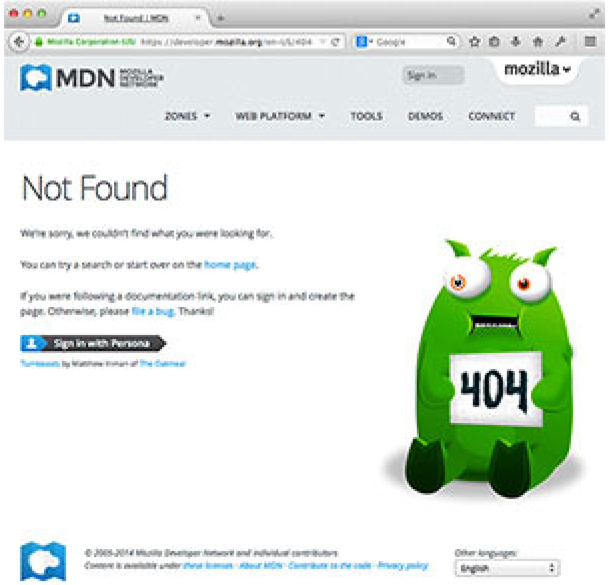
Neither the server nor the client remember previous communications. For example, relying on HTTP alone, a server can't remember a password you typed or remember your progress on an incomplete transaction. You need an application server for tasks like that. (We'll cover that sort of technology in other articles.)

HTTP provides clear rules for how a client and server communicate. We'll cover HTTP itself in a technical article later. For now, just be aware of these things:

- Only *clients* can make HTTP requests, and then only to *servers*. Servers can only *respond* to a *client*'s HTTP request.
- When requesting a file via HTTP, clients must provide the file's URL.
- The web server *must answer* every HTTP request, at least with an error message.

On a web server, the HTTP server is responsible for processing and answering incoming requests.

1. Upon receiving a request, an HTTP server first checks if the requested URL matches an existing file.
2. If so, the web server sends the file content back to the browser. If not, an application server builds the necessary file.



Not Found

We're sorry, we couldn't find what you were looking for.

You can try a search or start over on the home page.

If you were following a documentation link, you can sign in and create the page. Otherwise, please file a bug. Thanks!

Tumbeasts by Matthew Inman of The Oatmeal

© 2005-2014 Mozilla Developer Network and individual contributors
Content is available under these licenses · About MDN · Contribute to the code · Privacy policy

3. If neither process is possible, the web server returns an error message to the browser, most commonly `404 Not Found`. (The 404 error is so common that some web designers devote considerable time and effort to designing 404 error pages.)

Static vs. dynamic content

Roughly speaking, a server can serve either static or dynamic content. Remember that the term *static* means "served as-is". Static websites are the easiest to set up, so we suggest you make your first site a static site.

The term *dynamic* means that the server processes the content or even generates it on the fly from a database. This approach provides more flexibility, but the technical stack is more complex, making it dramatically more challenging to build a website.

Take, for example, the page you're reading right now. On the web server hosting it, there is an application server that takes article content from a database, formats it, puts it inside some HTML templates, and sends you the results. In this case, the application server is called Kuma. Kuma is built with the programming language Python (using the Django framework). The Mozilla team built Kuma for the specific needs of MDN, but there are many similar applications built on many other technologies.

There are so many application servers that it's difficult to suggest a particular one. Some application servers cater to specific website categories like blogs, wikis, or eCommerce; others, called CMSs (content management systems), are more generic. If you're building a dynamic website, take the time to choose technology that fits your needs. Unless you want to learn web server programming (which is an exciting area in itself!), you don't need to create your own application server. That's just Unknown prefix: Wikipedia..

Next steps

Now that you are familiar with web servers, you could:

- read up on how much it costs to do something on the web
- learn more about various software you need to create a website
- move on to something practical like how to upload files on a web server.

Last modified: Aug 6, 2020, by MDN contributors