Server-side scripting is a technique used in website design which involves embedding scripts in an HTML source code which results in a user's (client's) request to the server website being handled by a script running server-side before the server responds to the client's request. The scripts can be written in any of a number of server-side scripting languages available (see below). Server-side scripting differs from client-side scripting where embedded scripts, such as JavaScript, are run client-side in the web browser.

Server-side scripting is usually used to provide an interface and to limit client access to proprietary databases or other data sources. These scripts may assemble client characteristics for use in customizing the response based on those characteristics, the user's requirements, access rights, etc.

Server-side scripting also enables the website owner to reduce user access to the source code of server-side scripts which may be proprietary and valuable in itself. The down-side to the use of server-side scripting is that the server website computer needs to provide most of the computing resources before sending a page to the client computer for display via its web browser.

When the server serves data in a commonly used manner, for example according to the HTTP or FTP protocols, users may have their choice of a number of client programs (most modern web browsers can request and receive data using both of those protocols).

In the case of more specialized applications, programmers may write their own server, client, and communications protocol, that can only be used with one another.

Programs that run on a user's local computer without ever sending or receiving data over a network are not considered clients, and so the operations of such programs would not be considered client-side operations.

In the earlier days of the web, server-side scripting was almost exclusively performed by using a combination of C programs, Perl scripts, and shell scripts using the Common Gateway Interface (CGI). Those scripts were executed by the operating system, and the results were served back by the web server.

Many modern web servers can directly execute on-line scripting languages such as ASP and PHP either by the web server itself or via extension modules (e.g. mod_perl or mod php) to the web server. For example, WebDNA includes its own embedded database system. Either form of scripting (i.e., CGI or direct execution) can be used to build up complex multi-page sites, but direct execution usually results in lower overhead due to the lack of calls to external interpreters.

Dynamic websites sometimes use custom web application servers, for example the Python "Base HTTP Server" library, although some may not consider this to be server-side scripting. When designing using dynamic web-based scripting technics, like classic ASP or PHP, developers must have a keen understanding of the logical, temporal, and physical separation between the client and the server. For a user's action to trigger the execution of server-side code, for example, a developer working with classic ASP must explicitly cause the user's browser to make a request back to the web server. Creating such interactions can easily consume much development time and lead to unreadable code.