**Web Application Construction process**

## 3 Categories of Programming Languages

<https://www.killersites.com/blog/2006/categories-of-programming-languages/>

**Programming languages** are developers’ tools—and each is well suited for a particular kind of website, application type, or project size and scope. Developers will have their own preferences, and will also know which languages and frameworks to use to maximize an application’s potential as well as their own efficiency.

However, when it comes to these skills—probably the most overwhelming aspect for a nontechnical person to sift through—frameworks and languages are often front and center on developers’ resumes. That’s why we’ve compiled a quick look at the most popular programming languages and frameworks, so you can hire faster and more effectively.

### LANGUAGES FOR EVERY PART OF YOUR SITE: MARKUP LANGUAGES, CLIENT-SIDE SCRIPTS, AND SERVER-SIDE SCRIPTS

\*\*\* <https://www.upwork.com/hiring/development/how-scripting-languages-work/> **(important)**

Which area of development do you need support for? [*Client-side scripting*](https://www.upwork.com/hiring/development/how-scripting-languages-work/) creates what users interact with on your site; [*server-side scripting*](https://www.upwork.com/hiring/development/server-side-scripting-back-end-web-development-technology/) is typically your site’s back-end development; [*database technology*](https://www.upwork.com/hiring/data/a-guide-to-database-technology/) manages all the information on the server that supports a website; [*markup languages*](https://www.upwork.com/hiring/development/the-basics-of-web-development/) are the backbone of it all.

# Front-End Web Development: Client-Side Scripting & User Experience

\*\* <https://www.upwork.com/hiring/development/how-scripting-languages-work/>

\*\* <https://www.sqa.org.uk/e-learning/ClientSide01CD/page_01.htm>

Everything you see, click, and interact with on a website is the work of [front-end web development](https://www.upwork.com/hiring/development/how-do-you-hire-a-front-end-developer/). Client-side frameworks and scripting languages like [JavaScript](https://www.upwork.com/hiring/development/what-is-javascript/)and [AngularJS](https://www.upwork.com/hiring/development/angularjs-javascript-framework/) have made interactive websites possible. Here’s a look at how this technology works in the scheme of a website, and some of the most popular scripts and frameworks you should know.

### ****THE SERVER VS THE CLIENT****



All websites run on three components: **the server, the database,**and **the client.** The client is simply the browser a person is using to view a site, and it’s where client-side technology is unpacked and processed. *The server* is at a remote location anywhere in the world—housing data, running a site’s [back-end architecture](https://www.upwork.com/hiring/development/back-end-web-developer/), processing requests, and sending pages to the browser. *The client* is anywhere your users are viewing your site: **mobile devices, laptops,** or **desktop computers.**[Server-side scripting](https://www.upwork.com/hiring/development/server-side-scripting-back-end-web-development-technology/)is executed by a web server; **client-side scripting** is executed by a browser.

Client-end scripts are embedded in a website’s HTML [markup code](https://www.upwork.com/hiring/development/the-basics-of-web-development/), which is housed on the server in a language that’s compatible with, or compiled to communicate with, the browser. The browser temporarily downloads that code, and then, apart from the server, processes it. If it needs to request additional information in response to user clicks, mouse-overs, etc. (called “events”), a request is sent back to the server.

Client-side scripting is always evolving—it’s growing simpler, more nimble, and easier to use. As a result, sites are faster, more efficient, and less work is left up to the server.

# Server-Side Scripting: Back-End Web Development Technology

When you type in a URL, lots of code is at work to bring a page to your screen. What connects your site’s database to the browser, creating a smooth, user-friendly experience? That’s the software built by **server-side scripts,** languages that build your site behind the scenes. The goal of this software? To provide a seamless experience for the user that’s as close to a desktop application as possible.

There are many server-side languages working toward that end goal. The language you choose for your site depends on a mix of your site’s requirements, your database/operating system setup, and the preferences of your development team. Knowing what each script can offer and what sets it apart is helpful in deciding how to build your back end, and who to hire



The back end comprises three parts: the **server,** your [database](https://www.upwork.com/hiring/data/database-administrator-project-description/)**,** any **APIs**, and a **back-end web application,**software written via server-side languages. The server is a powerful computer that runs the back-end software, the database houses your site’s data, and the software communicates between the two. For example, if a user is updating a profile on a networking site, the server-side scripts will gather the information the user enters, the application will process it on the server, then interact with the database to update that information there.

Server-side scripts are used by [back-end web developers](https://www.upwork.com/hiring/development/back-end-web-developer/) to build the **back-end software**of a website—the mechanics we don’t see, but that make a site’s usability and functionality possible. These languages create the communication channel between user, server, and database. Anything that isn’t explicitly written into [the text markup of a site](https://www.upwork.com/hiring/development/the-basics-of-web-development/) is front-end or back-end software. Any data that a user requests in the browser (e.g., the fields in drop-down menus, photos, or user profiles) is delivered via server-side scripts, which create a channel between server and end user that requests, edits, and deletes things in the database. In the browser, [front-end scripts](https://www.upwork.com/hiring/development/how-scripting-languages-work/) make that information available to the user.

**Differences between Client-side and Server-side Scripting**

<https://www.sqa.org.uk/e-learning/ClientSide01CD/page_18.htm>

**Client-side Environment**

The client-side environment used to run scripts is usually a browser. The processing takes place on the end users computer. The source code is transferred from the web server to the users computer over the internet and run directly in the browser.

The scripting language needs to be **enabled** on the client computer. Sometimes if a user is conscious of **security risks** they may switch the scripting facility off. When this is the case a message usually pops up to alert the user when script is attempting to run.

**Server-side Environment**

The **server-side environment** that runs a scripting language is a web server. A user's request is fulfilled by running a script directly on the web server to generate dynamic HTML pages. This HTML is then sent to the client browser. It is usually used to provide interactive web sites that interface to databases or other data stores on the server.

This is different from client-side scripting where scripts are run by the viewing web browser, usually in JavaScript. The primary advantage to server-side scripting is the ability to highly customize the response based on the user's requirements, access rights, or queries into data stores.

#  Guide to Database Technology

## Databases store, organize, and process information in a way that makes it easy for us to go back and find what we’re looking for. We encounter databases, both simple and complex, all the time, whether in the form of library card catalogs, financial records, employee directories, and even phone books. But, what are databases in the context of a website? In this quick guide to modern database technology, you’ll get an understanding of how databases work, common terms to know, a look at [SQL vs. NoSQL](https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/), and how to determine which database is best for your web application.

# The Basics of Web Development: Markup Languages HTML, XML & XHTML

**Markup languages**are the brick and mortar of the Web—where it all started, when websites were just static pages with text and some formatting. Originating from typesetting processes used in early printing presses, these languages have long been used to annotate the text of a site, dictating both the architecture of a site and the display of text. While markup languages are a part of the past, that hasn’t made them obsolete. In fact, they’ve remained a core of development and its future as

Everything you see on the Web is a combination of **markup** (text), [CSS](https://www.upwork.com/hiring/development/css-cascading-style-sheets/)(design) and[front-end scripts](https://www.upwork.com/hiring/development/how-scripting-languages-work/)(interactivity), and that markup is what creates a site’s foundation. HTML is the main markup for web pages, or just about anything displayed in a browser, which explains why it’s still incredibly relevant, and why so many developers know it.

Here’s a look at how these languages fit into the fabric of web development as a whole, what makes them different, and a few of the most notable ones in use.