JavaScript

Chitradevi M
Technical Trainer
KGiSL Micro College
KGiSL Campus, Coimbatore – 641 035.

Static Websites

Responsive Websites

Dynamic Websites

What is a static website?

A static website is made up of webpages created using HTML, CSS and Javascript (all examples of web development languages). Each page on a static website is stored as a single HTML file, which is delivered directly from the server to the webpage exactly as is. This content essentially becomes a part of the design on your page, and won't change unless the original HTML file is edited at a code level.

Changes to a static website can be done manually, and will only be made page by page, HTML file by HTML file. For example, edits made to the HTML file of a homepage will only be reflected on the homepage. This is true even for elements that are identical across the whole site, such as the footer. If you're using a website builder, changes to static pages will be made automatically every time you use the website editor.

Advantages of a static website

Static sites are back—well, in some situations—and we’re going to go over several main reasons for this comeback.

- **Faster page loading speed**
- **Quick creation**
- **Potential for enhanced security**

Disadvantages of a static website

Having been through the benefits of a static site, we are now going to cover some of the disadvantages below.

- **Limited scalability**
- **Less efficient management**

What is a dynamic website?

Built using server side language and technology, dynamic websites allow for the content of each page to be delivered and displayed dynamically, or on-the-fly, according to user behavior or from user-generated content.

With a dynamic website all of your data and content are organized in a database or backend Content Management System (CMS), which connects to your website pages. The way this information is arranged and connected to your site’s design controls how and when its content is revealed on a page.

What does all of this mean? Well, dynamic content gives you the ability to customize and personalize the website experience, and what is displayed, for a specific user. It also allows you to make changes to many pages at the same time, since modifications made to one dynamic page can be automatically made across thousands.

For example, dynamic websites enable you to choose which information is displayed to a user based on their location. You can also deliver content to users based on their current or past actions on your site (thanks Cookies), which essentially means each visitor sees a different view of the content on a page. A multilingual website is a great example of when creating a dynamic website might be relevant.

Other examples of well-know dynamic websites include:

Instagram: as a social media site, dependent on user-generated content, Instagram relies on a dynamic website.

CNN: media outlets use dynamic websites to update their content, either in response to breaking news or as stories age.

Disney Plus: as a large streaming site, this dynamic entertainment website's dynamic nature allows its content to be chosen and displayed according to a user's location, subscription and preferences.

Advantages of a dynamic website

For many website creators dynamic pages are the only way to go, and for good reason. Dynamic pages have the following advantages:

- **Easily updated**
- **A better user experience**
- **Greater functionality**

Disadvantages of a dynamic website

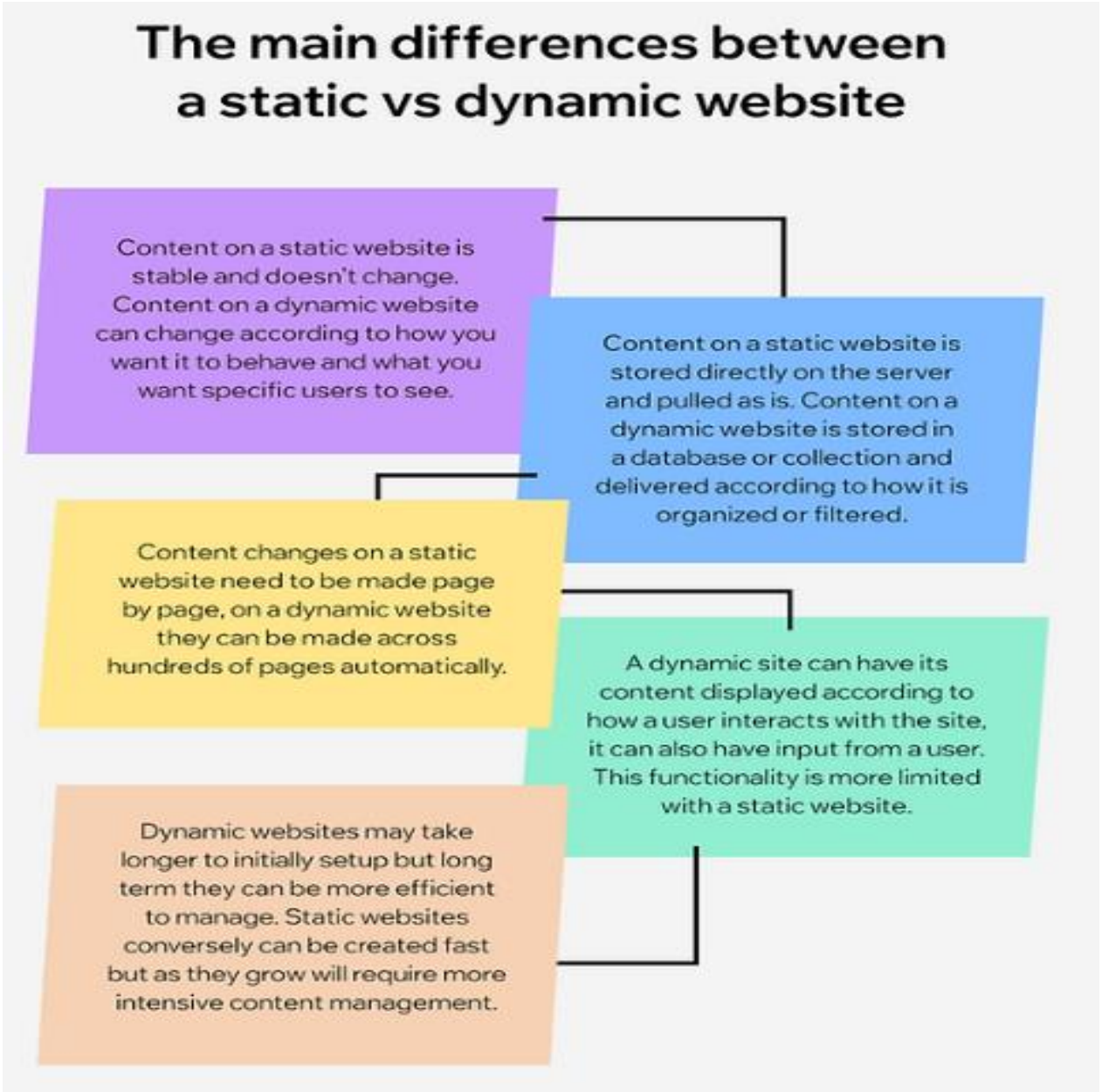
It takes more resources to create

Because of the extra steps needed to organize and connect your database to the right pages, a dynamic website can be more complicated to set up and get running . it will take more time to go live, and can be more costly, too.

Performance issues

Dynamic websites have more instructions to process than a static website does. They are also connected to a database or content collection and continually pull information from that in order to display it—which takes time to process and execute. This can impact the performance of a site, although many website creation tools are aware of this issue and make it their mission to prioritize performance across all pages.

The main differences between a static vs dynamic website

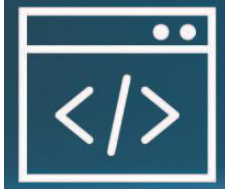


STATIC WEBSITE
VERSUS
DYNAMIC WEBSITE

STATIC WEBSITE	DYNAMIC WEBSITE
A website whose web pages are coded in HTML and the content of each page is fixed and does not change unless it is edited and republished	A website whose web pages are generated in real time
Developed using client-side technologies such as HTML and CSS	Deloped using client-side technologies as well as server-side scripting languages
Content remains unchanged unless it is changed from the source code	Content changes according to the client requests
Simple and easier to program	More complex and difficult to program
Does not allow many user interactions	Allow more user interaction
Do not access databases	Access information from a database
Cheaper to host	Costly to host
Difficult to update	Easier to update
Used for small-scale websites that do not require continuous changes	Suitable for large-scale e-commerce and social media websites

What Is a Programming Language?

A programming language
is a set of instructions that developers use
to create software, which users can in turn deploy
to interact with computers.

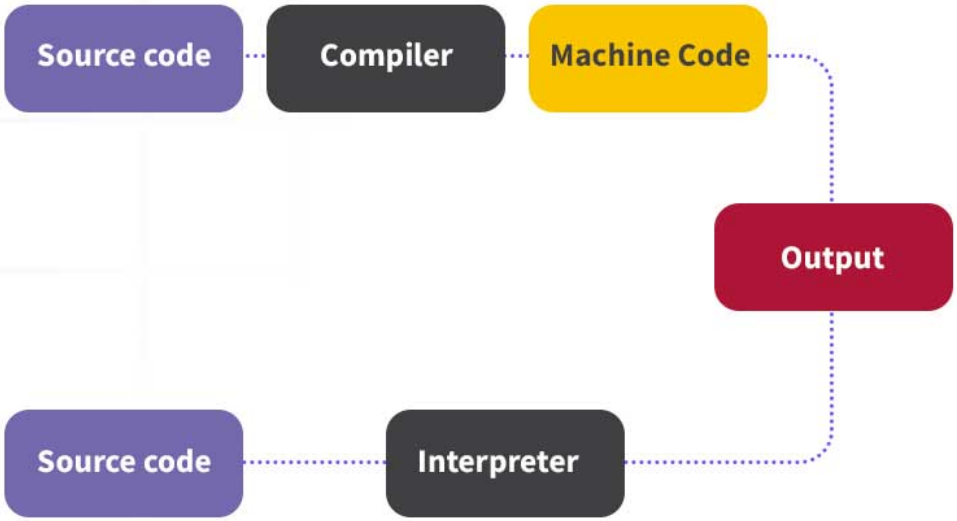


What is programming language

A programming language is a computer language programmers use to develop software programs, scripts, or other sets of instructions for computers to execute.

Although many languages share similarities, each has its own syntax. Once a programmer learns the languages rules, syntax, and structure, they write the source code in a text editor or IDE. Then, the programmer often compiles the code into machine language that can be understood by the computer. Scripting languages, which do not require a compiler, use an interpreter to execute the script.

What is **Scripting language?**



Scripting Language Definition

A scripting language (also known as scripting, or script) is a series of commands that can be executed without the need for compiling. While all scripting languages are programming languages, not all programming languages are scripting languages. PHP, Perl, and Python are common examples of scripting languages.

Scripting languages use a program known as an interpreter to translate commands and are directly interpreted from source code, not requiring a compilation step. Other programming languages, on the other hand, may require a compiler to translate commands into machine code before it can execute those commands.

COMPARISON BETWEEN

Programming Language

A programming language is an organized way of communicating with a computer.

Traditional programming is based on low level languages.

General programming leads to closed software applications.

More code needs to be written.

&

Scripting Language

A scripting language is a programming language that support scripts.

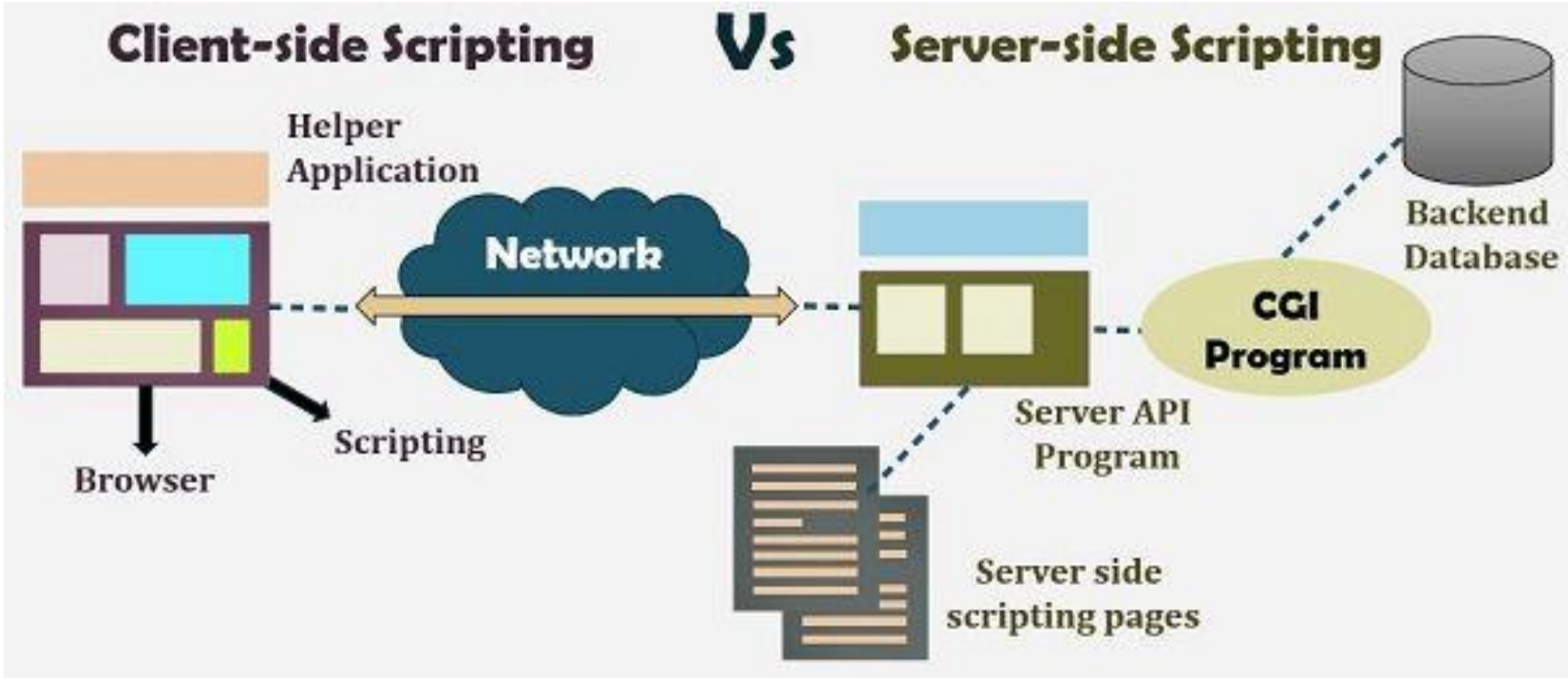
Scripting prefers high level languages.

Scripting promotes open projects and is used for web applications.

Less coding needs in scripting.

Interpreted vs Compiled Programming Languages

Understanding compiled vs interpreted languages is key to understanding how scripting languages function. An interpreted programming language is a language designed to execute source code directly and without the need to compile a program into machine-language instructions. An interpreter will execute the program by translating statements into a series of one or more subroutines before finally translating them into another language, such as machine code.


Client-side Scripting
Vs
Server-side Scripting
Helper Application
Network
Backend Database
CGI Program
Scripting
Browser
Server API Program
Server side scripting pages

scripting language is essential when entering a career in computer science or software engineering.

There are two types of scripting languages: **server side and client side.** The only significant difference between client-side scripting vs server-side scripting is that the former requires a server for its processing.

Server-side scripting languages run on a web server. When a client sends a request, the server responds by sending content via HTTP. In contrast, **client-side scripting languages** run on the client end, which means they run on their web browser.

Server-side scripting

Web servers are used to execute server-side scripting. They are basically used to create dynamic pages. It can also access the file system residing at the webserver. A server-side environment that runs on a scripting language is a web server.

Scripts can be written in any of a number of server-side scripting languages available. It is used to retrieve and generate content for dynamic pages. It is used to require to download plugins. In this load times are generally faster than client-side scripting. When you need to store and retrieve information a database will be used to contain data. It can use huge resources of the server. It reduces client-side computation overhead. The server sends pages to the request of the user/client.

Server-Side Scripting Languages

Language	Comments
PHP	The most popular server-side language used on the web.
ASP.NET	Web-application framework developed by Microsoft.
Node.js	Can run on a multitude of platforms, including Windows, Linux, Unix, Mac, etc.
Java	Used in everything from your car stereo's Bluetooth to NASA applications.
Ruby	Dynamic. Focuses heavily on simplicity.
Perl	A bit of a mashup between C, shell script, AWK, and sed.
Python	Great for beginners to learn. Uses shorter code.

Client-side scripting

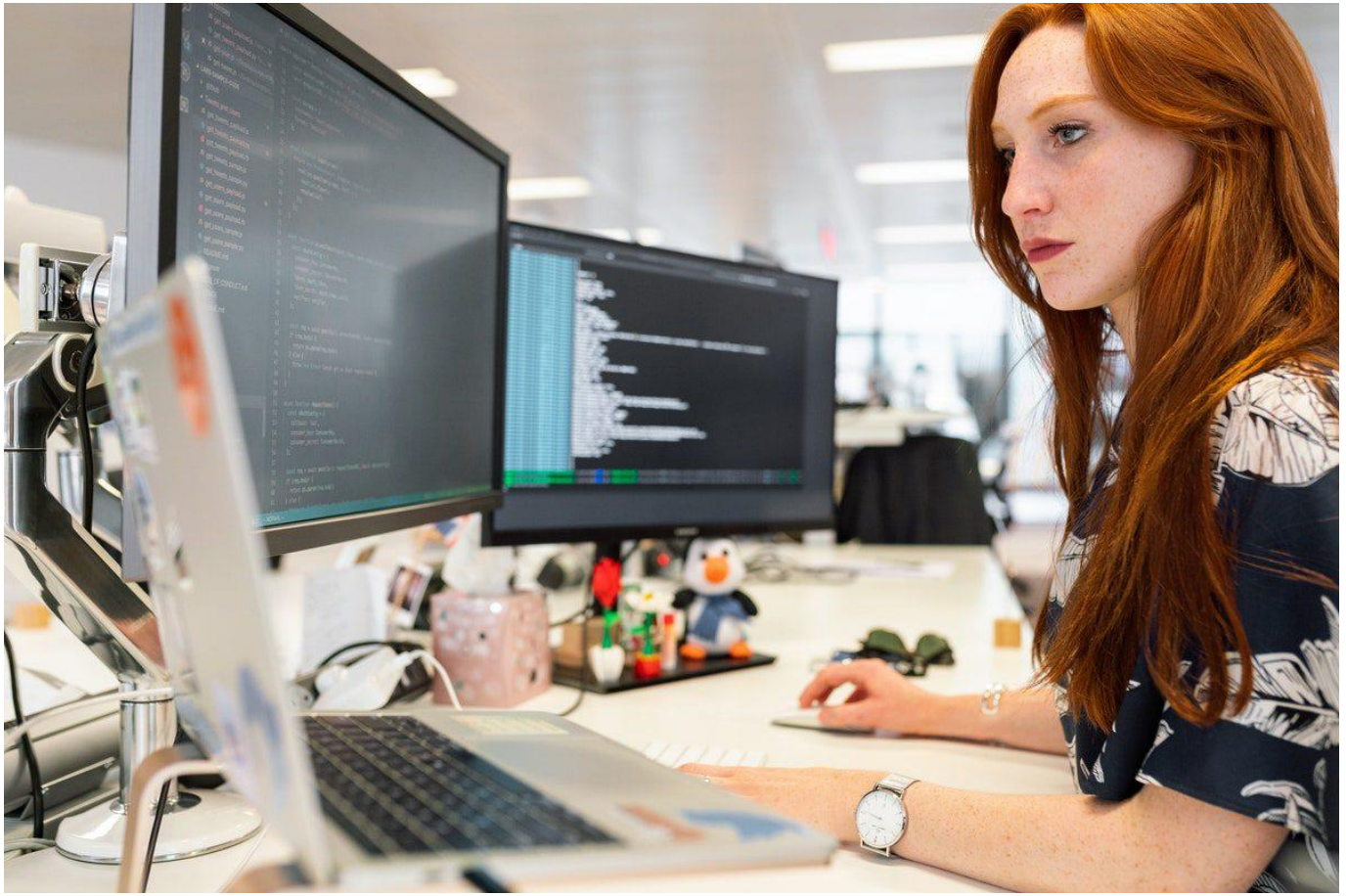
Web browsers execute client-side scripting. It is used when browsers have all code. Source code is used to transfer from webserver to user's computer over the internet and run directly on browsers. It is also used for validations and functionality for user events.

It allows for more interactivity. It usually performs several actions without going to the user. It cannot be basically used to connect to databases on a web server. These scripts cannot access the file system that resides in the web browser. Pages are altered on basis of the user's choice. It can also be used to create "cookies" that store data on the user's computer.

Client-Side Scripting Languages

Language	Comments
HTML	The foundation of web development.
CSS	Improves the appearance and graphics on pages in your web browser.
JavaScript	Though typically client side, can occasionally be used on the server side as well.

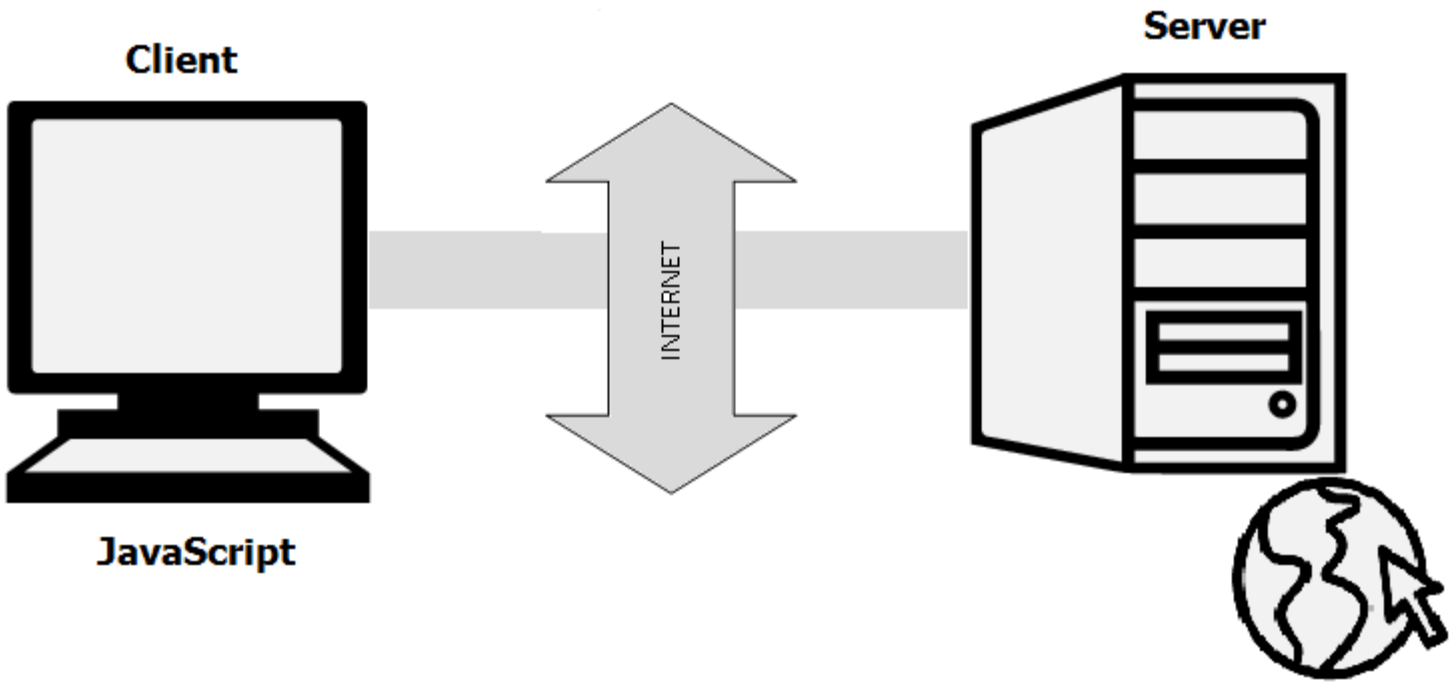
Server-Side Scripting vs Client-Side Scripting



Client-side scripting	Server-side scripting
Source code is visible to the user.	Source code is not visible to the user because its output of server-sideside is an HTML page.
Its main function is to provide the requested output to the end user.	Its primary function is to manipulate and provide access to the respective database as per the request.
It usually depends on the browser and its version.	In this any server-side technology can be used and it does not depend on the client.
It runs on the user's computer.	It runs on the webserver.
There are many advantages linked with this like faster. response times, a more interactive application.	The primary advantage is its ability to highly customize, response requirements, access rights based on user.
It does not provide security for data.	It provides more security for data.
It is a technique used in web development in which scripts run on the client's browser.	It is a technique that uses scripts on the webserver to produce a response that is customized for each client's request.
HTML, CSS, and javascript are used.	PHP, Python, Java, Ruby are used.
No need of interaction with the server.	It is all about interacting with the servers.
It reduces load on processing unit of the server.	It surge the processing load on the server.

What is JavaScript

JavaScript is a very powerful **client-side scripting language**. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage more lively and interactive, with the help of JavaScript. JavaScript is also being used widely in game development and Mobile application development.


Client
Server
INTERNET
JavaScript

Dynamic Web Applications

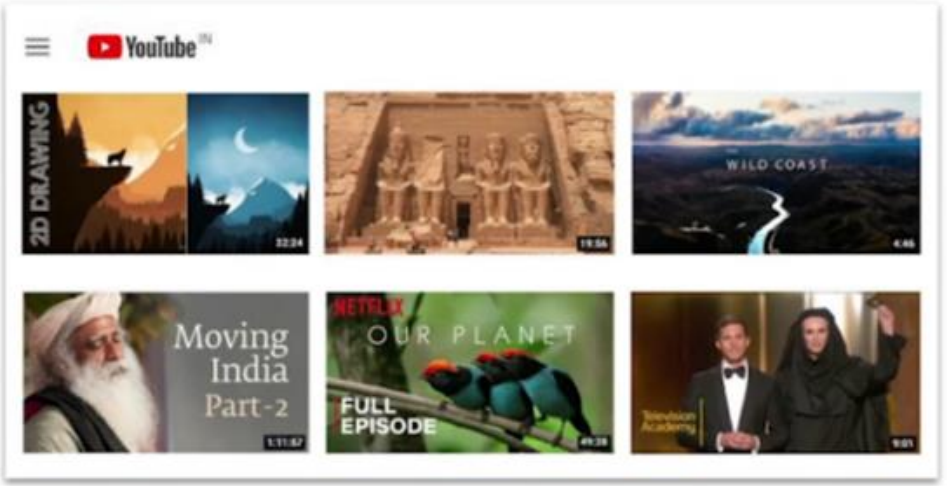
Personalised Content

User Specific

Instagram Feed

User Specific

Video Recommendations



Real-Time

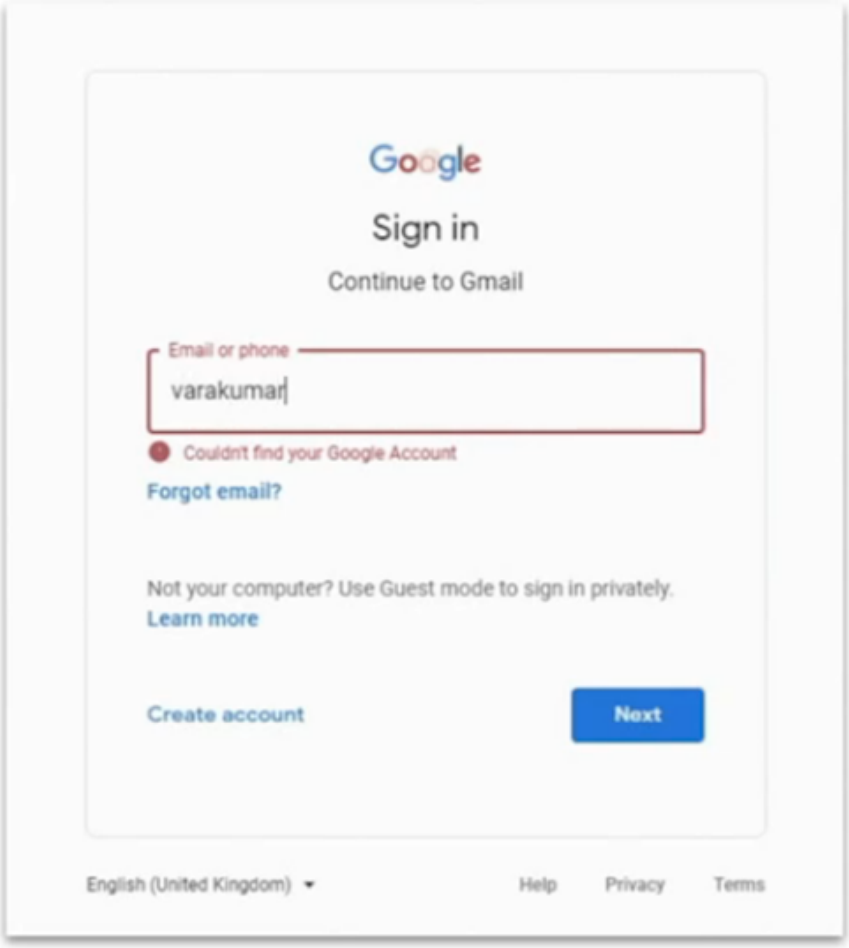
Live

Cricket Score Updates

User Interactivity

Form

Data Validations



Introduction to JavaScript

JavaScript is a scripting language that is used to create and manage dynamic web pages, basically anything that moves on your screen without requiring you to refresh your browser. It can be anything from animated graphics to an automatically generated Facebook timeline.

When most people get interested in web development, they start with good old HTML and CSS. From there, they move on to JavaScript, which makes sense, because, these three elements together form the backbone of web development.

•HTML is the structure of your page like the headers, the body text, any images you want to include. It basically defines the contents of a web page.

•CSS controls how that page looks (it’s what you’ll use to customize fonts, background colors, etc.).

•JavaScript is the third element. Once you’ve created your structure (HTML) and your aesthetic vibe (CSS), JavaScript makes your site dynamic (automatically updateable).

Javascript History

JavaScript was developed by Brendan Eich in 1995, which appeared in Netscape, a popular browser of that time.

Brendan Eich - Creator of JavaScript

The language was initially called LiveScript and was later renamed JavaScript. There are many programmers who think that JavaScript and Java are the same. In fact, **JavaScript and Java are very much unrelated. Java is a very complex programming language whereas JavaScript is only a scripting language**. The syntax of JavaScript is mostly influenced by the programming language C.

Why JavaScript?

- ✓ Javascript is the most popular programming language in the world and that makes it a default choice for web development. There are many frameworks available which you can use to create web applications once you have learned JavaScript.

- ✓ JavaScript offers lots of flexibility. You can create stunning and fast web applications with tons of customizations to provide users with the most relevant graphical user interface.

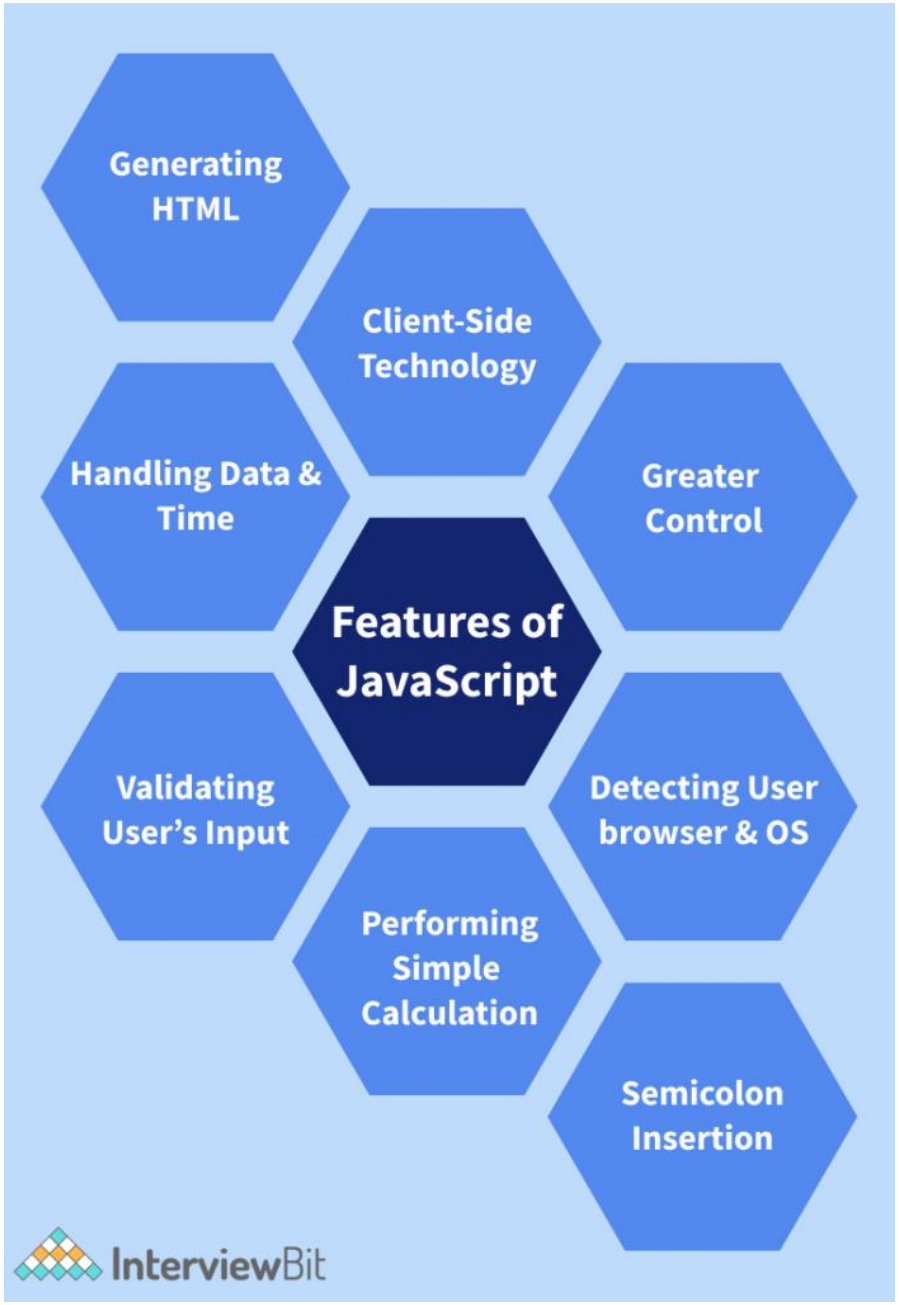
- ✓ JavaScript is now also used in mobile app development, desktop app development, and game development. This opens many possibilities for you as a Javascript developer.

- ✓ Due to the high demand in the industry, there are tons of job growth opportunities and high pay for those who know JavaScript.
- ✓ The incredible thing about Javascript is that you can find tons of frameworks and libraries already developed, which can be used directly in web development. That reduces the development time and enhances the graphical user interface.

What is JavaScript Used For?

JavaScript is used in various fields from the web to servers, and here's a quick list of the significant areas it's used in:

- ✓ Web Applications: JavaScript is used for adding interactivity and automation to websites. So, if you want your web application to be anything more than just a static page of contents, you'll probably need to do some "JavaScript'ing."
- ✓ Mobile Applications: JavaScript isn't just for developing web applications; it is also used for developing applications for phones and tablets. With frameworks like React Native, you can develop full-fledged mobile applications with all those fancy animations.
- ✓ Web-based Games: If you've ever played a game directly on the web browser, JavaScript was probably used to make that happen.
- ✓ Back-end Web Development: JavaScript has traditionally been used for developing the front-end parts of a web application. However, with the introduction of NodeJS, a prevalent back-end JavaScript framework, things have changed. And now, JavaScript is used for developing the back-end structure also.


Generating HTML
Client-Side Technology
Handling Data & Time
Greater Control
Features of JavaScript
Validating User's Input
Detecting User browser & OS
Performing Simple Calculation
Semicolon Insertion
InterviewBit

How to Run JavaScript?

Being a scripting language, **JavaScript cannot run on its own. In fact, the browser is responsible for running JavaScript code**. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it is up to the browser to execute it. The main advantage of JavaScript is that **all modern web browsers support** JavaScript. So, you do not have to worry about whether your site visitor uses Internet Explorer, Google Chrome, Firefox or any other browser. JavaScript will be supported. Also, JavaScript **runs on any operating system** including Windows, Linux or Mac. Thus, JavaScript overcomes the main disadvantages of VBScript (Now deprecated) which is limited to just IE and Windows.

Tools You Need

To start with, you need a text editor to write your code and a browser to display the web pages you develop. You can use a text editor of your choice including Notepad++, Visual Studio Code, Sublime Text, Atom or any other text editor you are comfortable with. You can use any web browser including Google Chrome, Firefox, Microsoft Edge, Internet Explorer etc.

Sublime Text

Bluefish

Visual Code Editor

TextMate

TextWrangler


PLATFORM:
Mac
Windows
Linux
Android
BROWSERS:
Chrome
Firefox
Safari
Opera
IE
WEB BROWSERS

A Simple JavaScript Program

JavaScript code within **<script> tags** (<script> and </script>) if you are keeping your JavaScript code within the HTML document itself. This helps your browser distinguish your JavaScript code from the rest of the code. As there are other client-side scripting languages (Example: VBScript), it is highly recommended that you specify the scripting language you use. You have to use the type attribute within the <script> tag and set its value to text/javascript like this:

<script type="text/javascript">

JavaScript example is easy to code. JavaScript provides 3 places to put the JavaScript code:

within body tag, within head tag and external JavaScript file.

The **script** tag specifies that we are using JavaScript.

The **text/JavaScript** is the content type that provides information to the browser about the data.

The **document.write()** function is used to display dynamic content through JavaScript.

3 Places to put JavaScript code

Between the body tag of html

Between the head tag of html

In .js file (external JavaScript)

Between the head tag of html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
   <script type="text/javascript">
      alert("Hello");
   </script>
</head>
<body>
 </body>
</html>
```

Between the body tag of html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>

</head>
<body>
   <script type="text/javascript">
      alert("Hello");
   </script>
</body>
</html>
```

External JavaScript file

We can create external JavaScript file and embed it in many html page.
It provides **code re usability** because single JavaScript file can be used in several html pages.
An external JavaScript file must be saved by .js extension. It is recommended to embed all JavaScript files into a single file. It increases the speed of the webpage.

Index.html

```
<html>
<head>
<script type="text/javascript" src="msg.js"></script>
</head>
<body>

</body>
</html>
```

Script.js

```
alert("Hello Kgisl");
```

Building Dynamic Web Applications mainly involves

Client Server Communication

Manipulating HTML and CSS

Writing Application Logic

Client-side validation,

Dynamic drop-down menus,

Displaying date and time,

Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),

Displaying clocks etc.

JavaScript Output

innerHTML

document.write()

Window.alert()

Console.log()

Window.prompt()

innerHTML

innerHTML: It is used to access an element. It defines the HTML content.

Syntax:

```
document.getElementById("id").innerHTML;
```

```
<h2>
JavaScript Display Possibilities
Using innerHTML
</h2>
<p id=“paraid"></p>

<!-- Script to use innerHTML -->

<script>
document.getElementById(" paraid ").innerHTML= 10 * 2;
</script>
```

document.write():

document.write(): It is used for testing purpose.

Syntax:

```
document.write()
```

```
<h2>
        JavaScript Display Possibilities
        Using document.write()
</h2>
<p id=“pid"></p>
<!-- Script to uses document.write() -->
<script>
        document.write(10 * 2);
        document.write(“Welcome to KGiSL Microcollege”);

</script>
```

window.alert()

window.alert(): It displays the content using an alert box.

Syntax:
window.alert()

```
<h2>
        JavaScript Display Possibilities
        Using window.alert()
</h2>
<p id=“pid"></p>
<!-- Script to use window.alert() -->
<script>
        window.alert(“Hai Students”);
</script>
```

console.log()

console.log(): It is used for debugging purposes.

Syntax:
console.log()

console.log()

```
<h2>
        JavaScript Display Possibilities
        Using console.log()
</h2>
<p id=“pid"></p>
<!-- Script to use console.log() -->
<script>
        console.log(10*2);
</script>
```

window.prompt()

window.prompt() :- it Allows to take input from user

syntax :
window.prompt()

```
<h2>
        JavaScript Display Possibilities
        Using window.alert()
</h2>
<p id=“pid"></p>
<!-- Script to use window.alert() -->
<script>
        window.prompt("Please Enter your Input");
</script>
```

Developer Tools

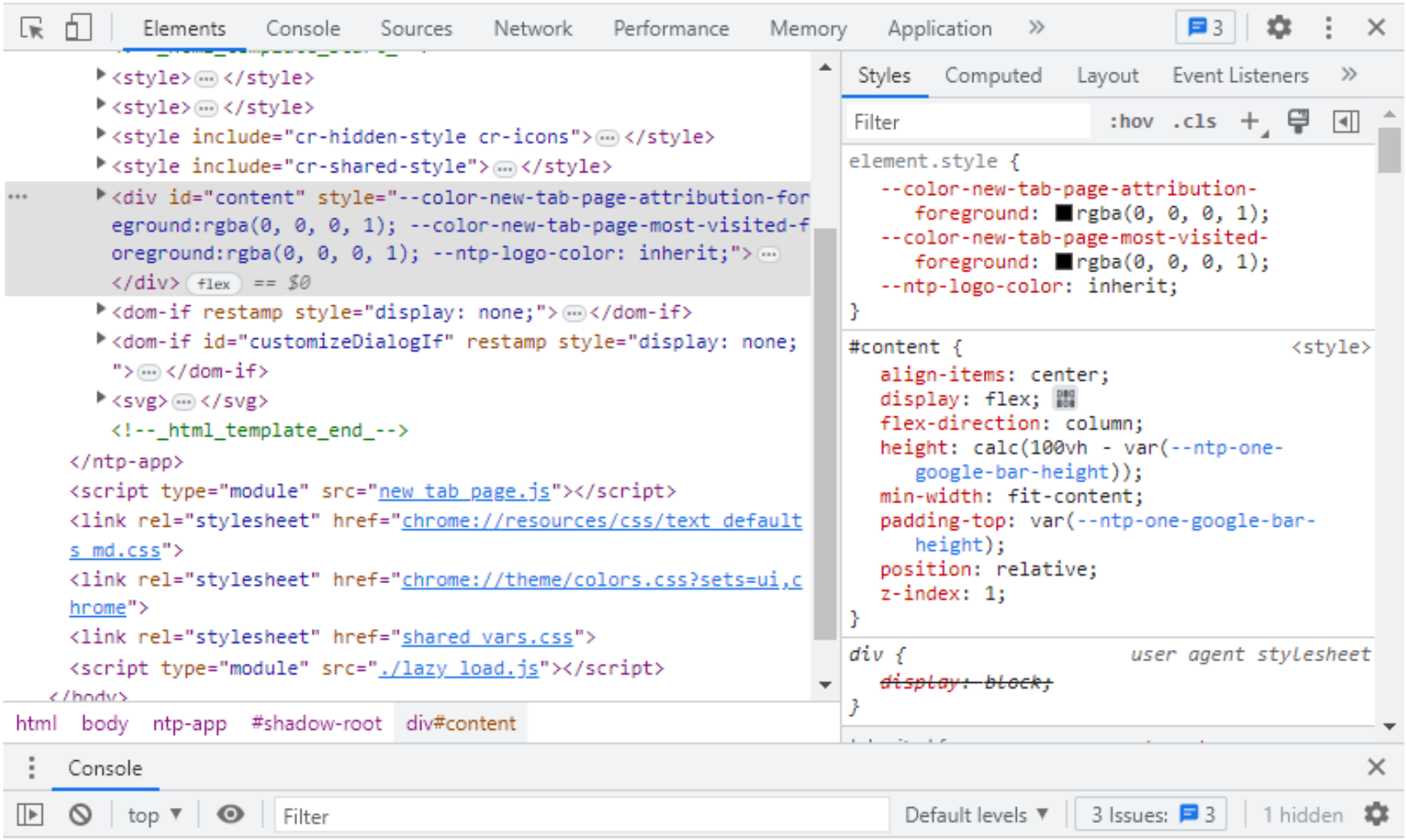
Chrome Developer Tools is a comprehensive toolkit for developers, built directly into the Chrome browser. These tools let you edit web pages in real time, diagnose problems more quickly, and build better websites faster.
Even if you’re not a developer, though, you can still get some use out of Developer Tools.

What are Chrome Developer Tools?

- Elements
- Console
- Source
- Network
- Application
- Security
- Memory
- Performance
- Audits

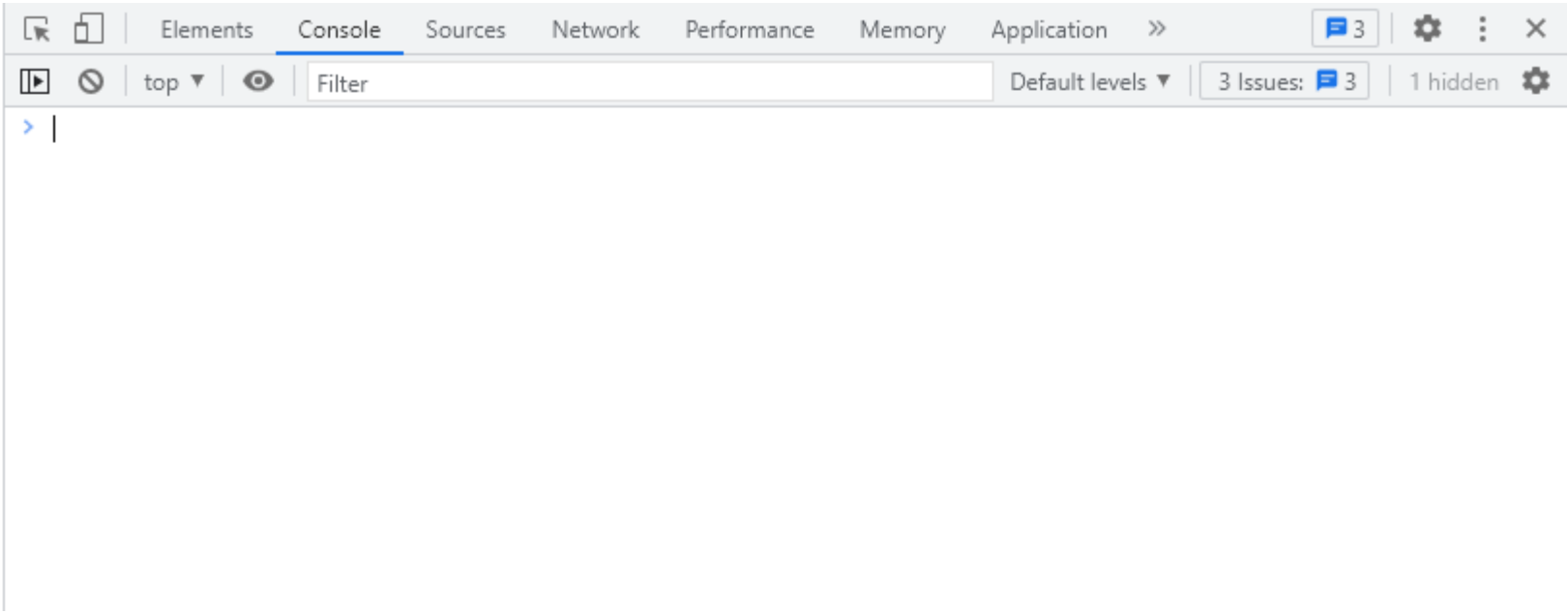
Elements

Elements shows you the HTML used to build the page you're looking at, together with any inline CSS.


Elements
Console
Sources
Network
Performance
Memory
Application
3
<style>…</style>
<style>…</style>
<style include="cr-hidden-style cr-icons">…</style>
<style include="cr-shared-style">…</style>
… flex == $0
<dom-if restamp style="display: none;">…</dom-if>
<dom-if id="customizeDialogIf" restamp style="display: none;">…</dom-if>
<svg>…</svg>
<!--_html_template_end_-->
</ntp-app>
<script type="module" src="new_tab_page.js"></script>
<link rel="stylesheet" href="chrome://resources/css/text_defaults_md.css">
<link rel="stylesheet" href="chrome://theme/colors.css?sets=ui,chrome">
<link rel="stylesheet" href="shared_vars.css">
<script type="module" src="./lazy_load.js"></script>
html
body
ntp-app
#shadow-root
div#content
Styles
Computed
Layout
Event Listeners
Filter
:hov
.cls
element.style {
--color-new-tab-page-attribution-foreground: rgba(0, 0, 0, 1);
--color-new-tab-page-most-visited-foreground: rgba(0, 0, 0, 1);
--ntp-logo-color: inherit;
}
#content {
<style>
align-items: center;
display: flex;
flex-direction: column;
height: calc(100vh - var(--ntp-one-google-bar-height));
min-width: fit-content;
padding-top: var(--ntp-one-google-bar-height);
position: relative;
z-index: 1;
}
div {
user agent stylesheet
display: block;
}
Console
top
Filter
Default levels
3 Issues: 3
1 hidden
>

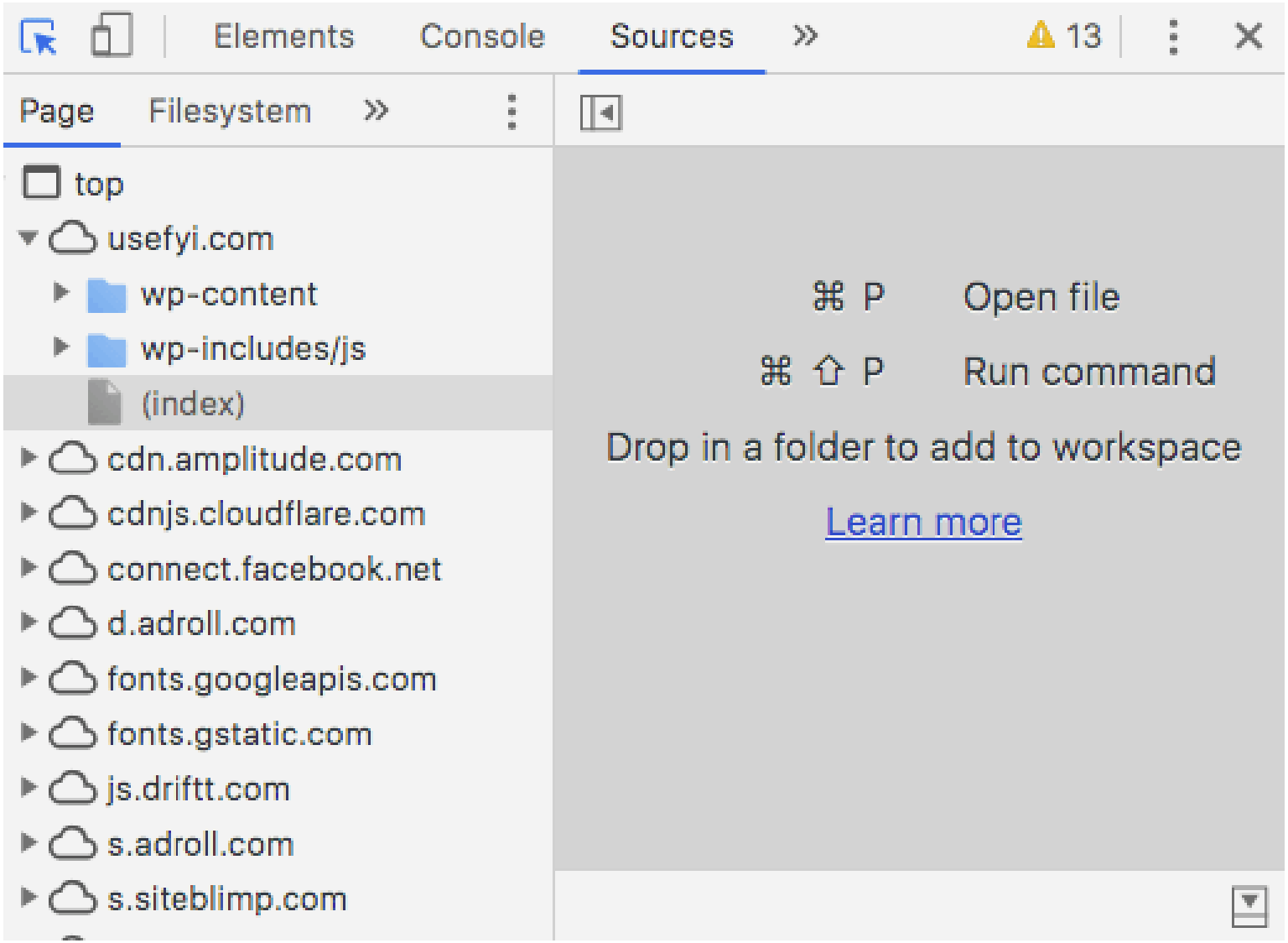
Console

Console deals with JavaScript. It gives you information about interactive elements on a page. In Console, you can write JavaScript to interact with the web page you're viewing, and it also lets you write messages to yourself in the JavaScript of websites you're building, which then show up in Console to show that the JS was executed.


Elements
Console
Sources
Network
Performance
Memory
Application
3
top
Filter
Default levels
3 Issues: 3
1 hidden

Sources

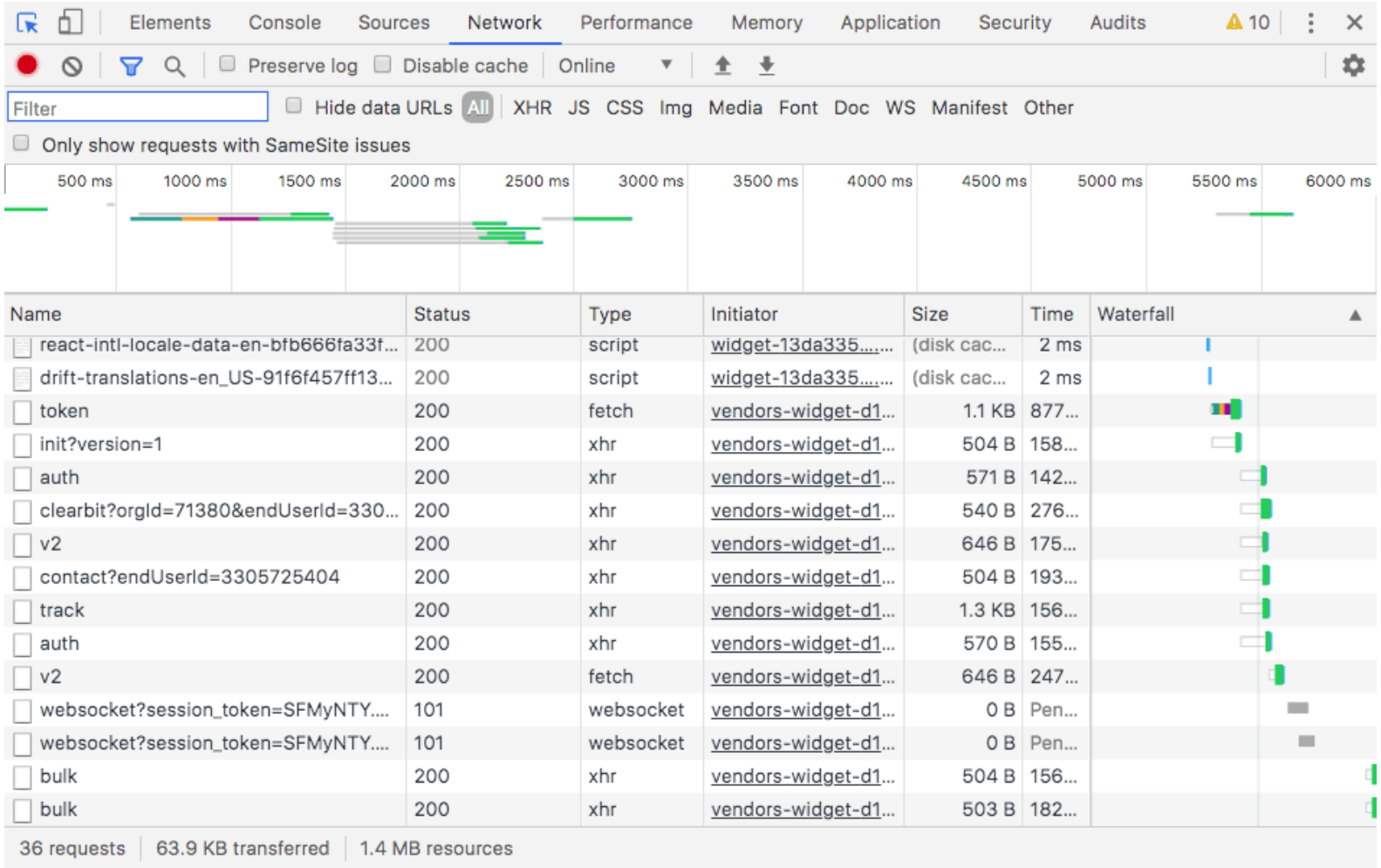
The **Sources** tab shows you where all the files that were used to make the website are stored and lets you inspect them.


Elements
Console
Sources
13
Page
Filesystem
top
usefyi.com
wp-content
wp-includes/js
(index)
cdn.amplitude.com
cdnjs.cloudflare.com
connect.facebook.net
d.adroll.com
fonts.googleapis.com
fonts.gstatic.com
js.driftt.com
s.adroll.com
s.siteblimp.com
⌘ P Open file
⌘ ⇧ P Run command
Drop in a folder to add to workspace
Learn more

Network

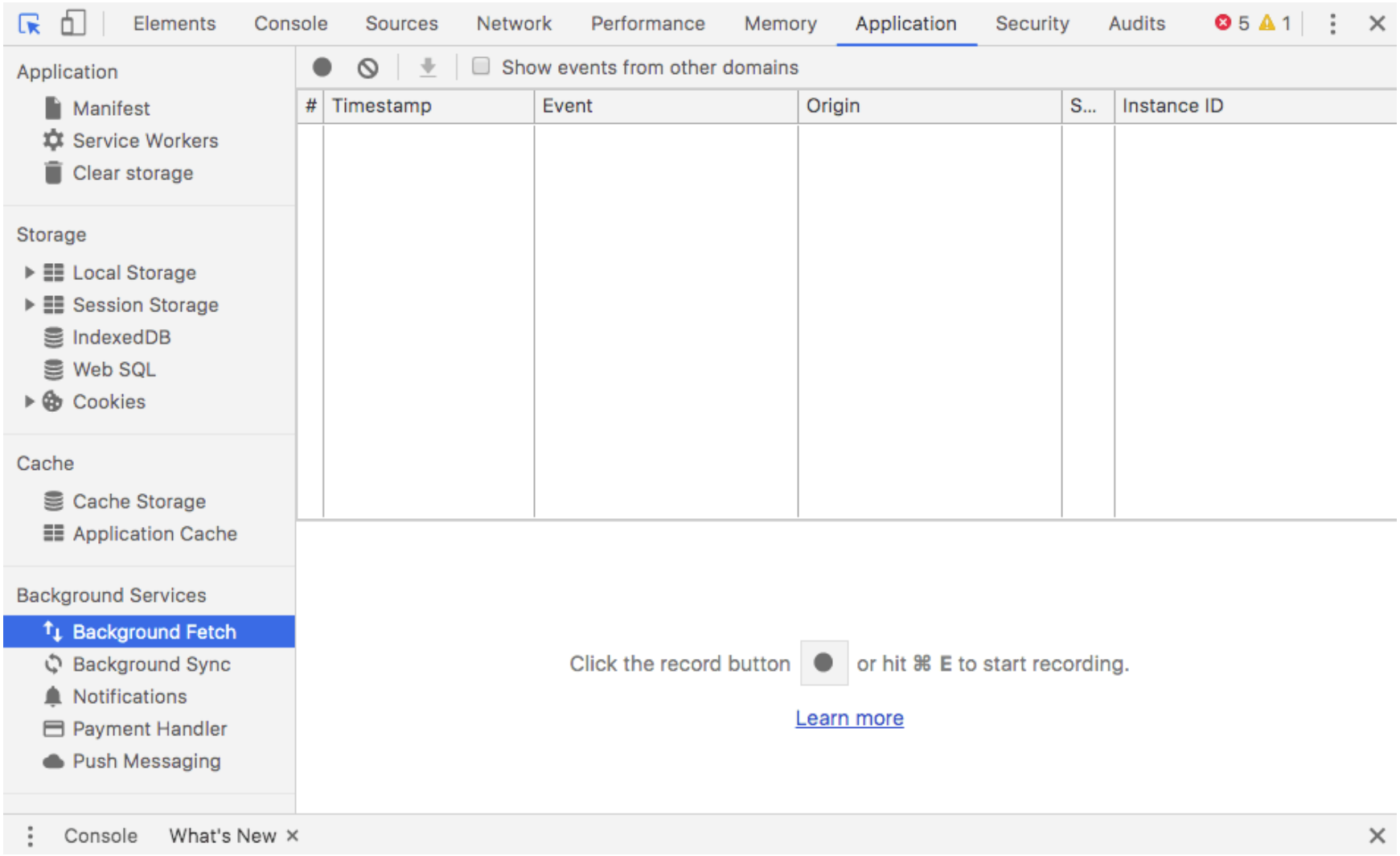
The **Network** tab shows you all the files that are loading in the URL you’re looking at.

You get a waterfall and deep data on all the items loaded, including initiator and time to load that element.



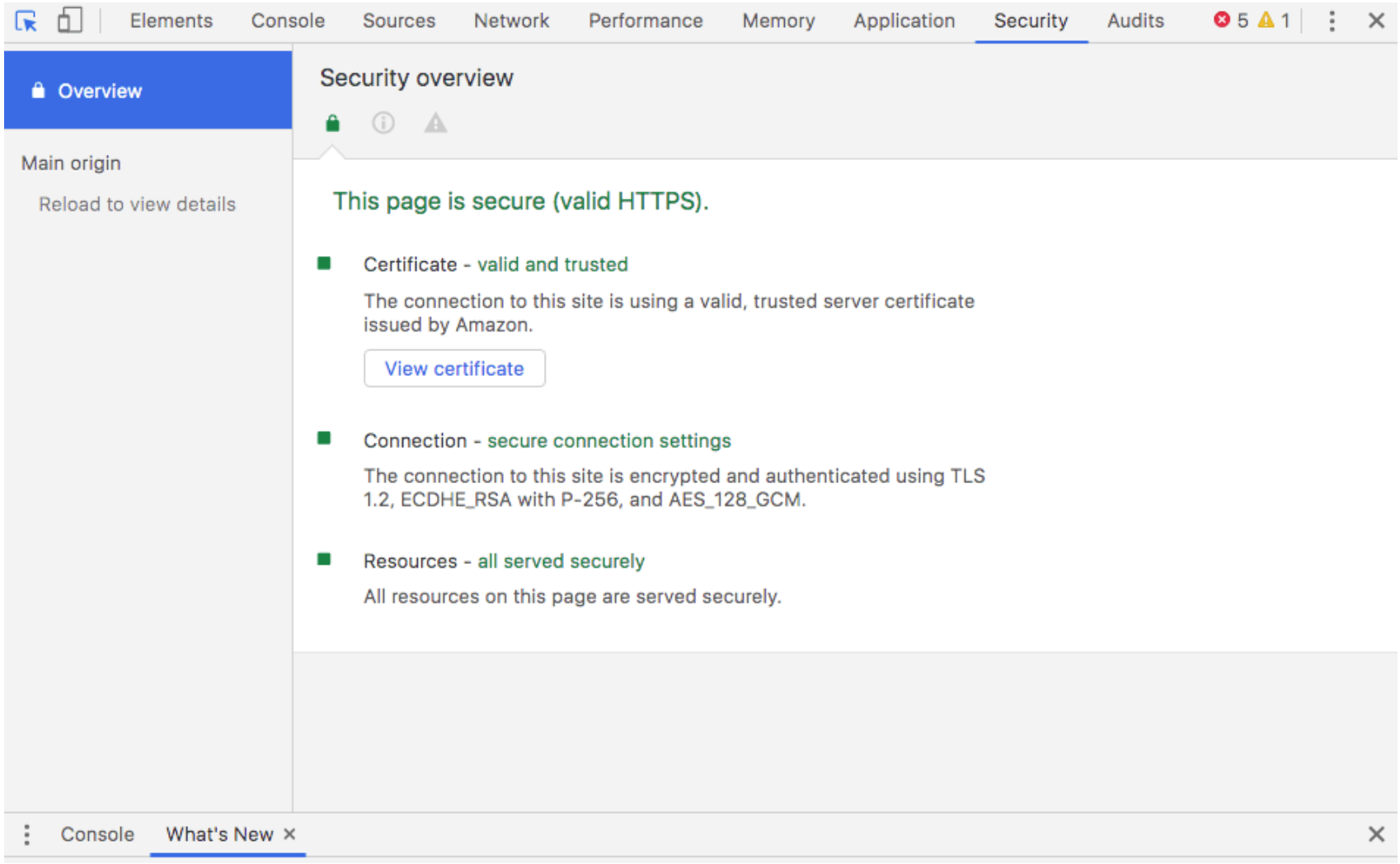
Application

Application shows you what’s in your browser storage: in-browser databases like Web SQL, local storage, and more. It also gives you granular control over your cookies.


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
5
1
Application
Manifest
Service Workers
Clear storage
Storage
Local Storage
Session Storage
IndexedDB
Web SQL
Cookies
Cache
Cache Storage
Application Cache
Background Services
Background Fetch
Background Sync
Notifications
Payment Handler
Push Messaging
Show events from other domains
#
Timestamp
Event
Origin
S...
Instance ID
Click the record button
or hit ⌘ E to start recording.
Learn more
Console
What's New

Security

Security gives you basic security information, letting you view a site's HTTPS certificate and TLS status.


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
5
1
Overview
Main origin
Reload to view details
Security overview
This page is secure (valid HTTPS).
Certificate - valid and trusted
The connection to this site is using a valid, trusted server certificate issued by Amazon.
View certificate
Connection - secure connection settings
The connection to this site is encrypted and authenticated using TLS 1.2, ECDHE_RSA with P-256, and AES_128_GCM.
Resources - all served securely
All resources on this page are served securely.
Console
What's New