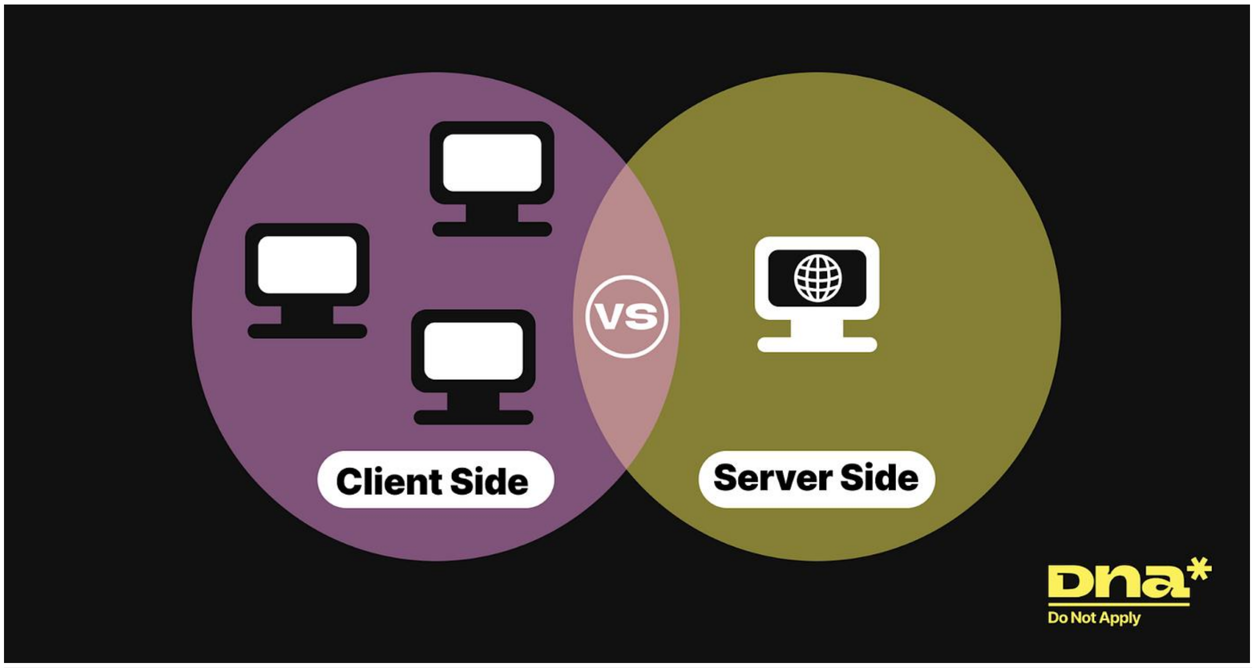
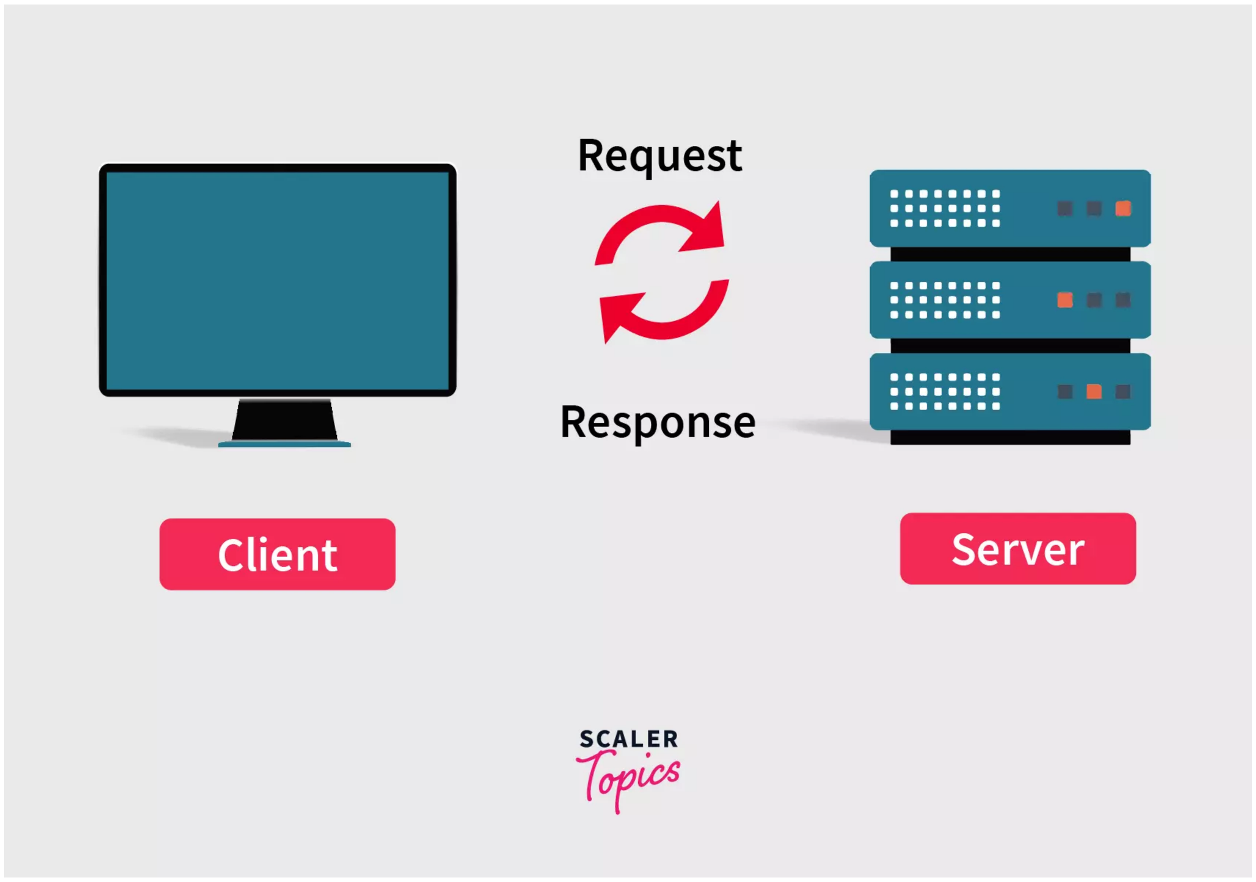
Unit 6.

Introduction to Server Side and Client-Side Scripting

- Overview of Server Side and Client-Side Scripting
- Difference between Server Side and Client-Side Scripting
- Advantages and Disadvantages of Server Side and Client-Side Scripting


VS
Client Side
Server Side
Dna*
Do Not Apply


Request
Response
Client
Server
SCALER
Topics

Script, Scripting and Scripting language

Script:

- A script is a set of instructions or commands given to the computer.
- It is written in a programming language.
- Scripts are easy to read and write.
- Scripts are used to automate tasks, perform calculations, manipulate data, or control the behavior of a software program etc.

Scripting:

Scripting refers to the process of writing or creating scripts.

Scripting Language:

- A scripting language is a programming language for writing scripts.
- Scripting languages typically use interpreter rather than compiler.
- It is easy to learn and use, which makes them a good choice for beginners.
- Here are some examples of scripting languages:
  - JavaScript
  - Python
  - PHP
  - Ruby
  - Perl
- Scripting languages are used in a wide variety of applications, including:
  - Web development
  - System administration
  - Data processing
  - Scientific computing
  - Game development
  - Multimedia development

The scripts may be created in ***two*** ways: on the ***client side*** or the ***server side*** .

Client-side scripting:

- Client-side scripting refers to the execution of scripts on the client's computer, typically in the user's web browser.
- It adds interactivity to web pages.
- Examples include form validation, animations, and interactive elements.
- JavaScript, VBScript, and ActionScript are common client-side scripting languages.
- Scripts are downloaded with HTML from the server.
- Browsers execute the scripts.
- It enables dynamic and interactive web pages.
- It handles validations and user event functionality.
- It can't directly connect to web server databases or access the browser's file system.
- It can modify pages based on user choices and create cookies to store data on the user's computer.

Advantages of client-side scripting

1. More interactivity: Client-side scripting makes web pages more engaging for users.
2. Less server load: Client-side scripting reduces the server's tasks by handling validation and animation on the user's computer.
3. Better security: Client-side scripting protects sensitive data by keeping it on the user's machine instead of sending it to the server.
4. Instant feedback: Client-side scripting provides immediate responses to user actions, improving the website's responsiveness.
5. Improved scalability: Client-side scripting lightens the server's workload, allowing for better scalability.
6. Enhanced user experience: Client-side scripts enable dynamic content updates without page refreshing, creating a smoother and more enjoyable user experience

Disadvantages of client-side scripting

1. Security risks: Client-side scripts can be targeted by security attacks like cross-site scripting (XSS), exposing vulnerabilities.
2. Browser compatibility challenges: Developers need to ensure client-side scripts work across different web browsers, which can be a complex task.
3. Performance impact: Client-side scripts may cause slower page loading, especially on older or slower computers, affecting performance.

4. Restricted control: Client-side scripts have limited access to server-side resources, making complex operations and database interactions more challenging.

Examples of client-side scripting

- Validation: Client-side scripting can be used to validate form input, such as ensuring that a user enters a valid email address or phone number.
- Animation: Client-side scripting can be used to add animation to web pages, such as a spinning logo or a scrolling banner.
- Dynamic content updates: Client-side scripting can be used to update the content of a web page dynamically, such as displaying the current time or weather conditions.

Server-side scripting

Server-side scripting refers to the execution of scripts on the server, which is the computer that hosts the web page.

- Server-side scripting executes scripts on the web server to generate dynamic content.
- Common server-side scripting languages are PHP, ASP.NET, and Ruby on Rails.
- Server-side scripts are written in languages like PHP, Python, Ruby, or Java.
- Server-side scripting involves processing form data, database operations, and server-side validation.
- Web servers are used to execute server-side scripting and create dynamic pages.
- Server-side scripting can access the web server's file system.
- It is used to retrieve and generate content for dynamic pages.
- Server-side scripting can reduce client-side computation overhead.
- It uses a database to store and retrieve information.
- The server sends pages to the user/client upon request.

Advantages of server-side scripting

- Full control: Server-side scripts run on the server, which gives developers complete control over the environment.
- Data security: Server-side scripts can access and process sensitive data on the server, which is more secure than storing it on the client's computer.
- Scalability: Server-side scripts can be scaled to handle large numbers of requests.

- Cross-browser compatibility: Server-side scripts generate HTML that is sent to the client, ensuring consistent behavior across different browsers.

- Access to server resources: Server-side scripts have direct access to server resources like databases, file systems, and external APIs, enabling complex operations and data manipulation.

Disadvantages of server-side scripting

- Increased server load: Server-side scripts can increase the load on the server, which can lead to slower performance.
- Complexity: Server-side scripting can be more complex than client-side scripting, which can make it more difficult to develop and maintain.
- Security: Server-side scripts are more vulnerable to security attacks than client-side scripts.
- Limited interactivity: Server-side scripts generate HTML that is sent to the client, so they cannot provide immediate feedback or interact with the user without additional client-side scripting.

Examples of server-side scripting

- Generating dynamic content: Server-side scripting can be used to generate dynamic content, such as search results, product listings, and user profiles.
- Processing data: Server-side scripting can be used to process data, such as calculating shipping costs or updating a database.
- Controlling access: Server-side scripting can be used to control access to web pages, such as requiring users to log in before they can view certain content.

.

Client-scripting vs Server-side scripting

Client-side scripting	Server-side scripting
Source code is visible to the user.	Source code is not visible to the user.
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.	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 is used.
No need of interaction with the server.	It is all about interacting with the servers.

Client-side scripting	Server-side scripting
It reduces load on processing unit of the server.	It increases the processing load on the server.