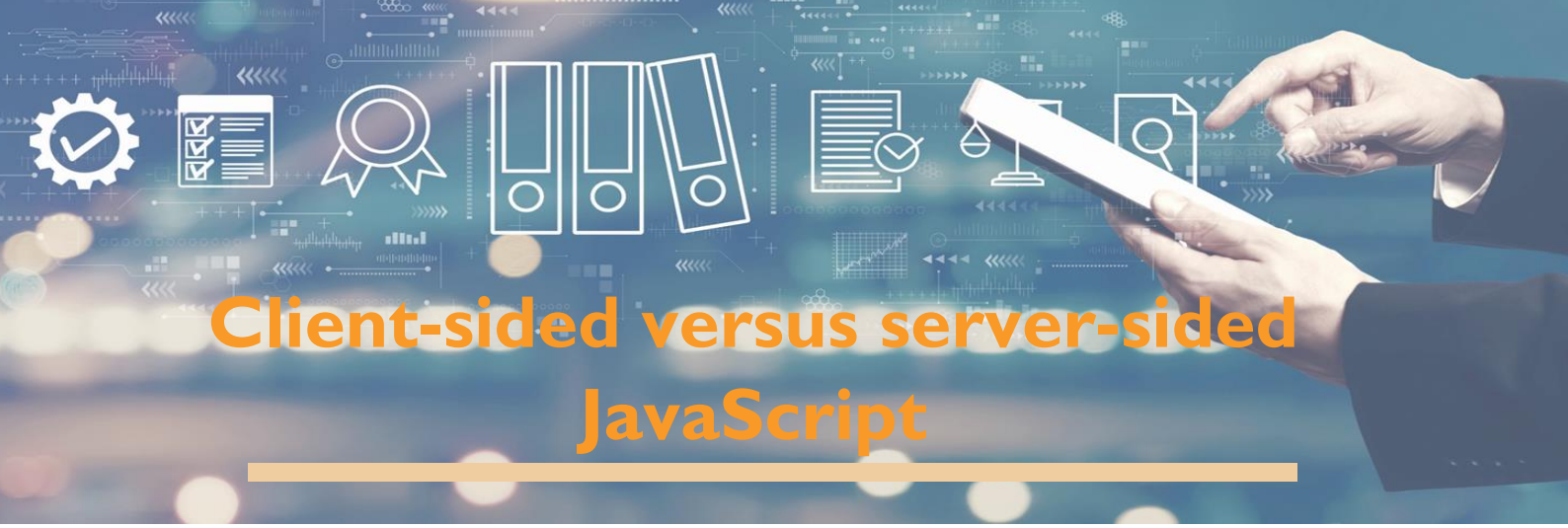

Client-sided versus server-sided JavaScript

Server-side vs. client-side applicaties

De focus van deze cursus ligt op client-sided JavaScript. Hier wordt voornamelijk mee bedoeld dat websites met JavaScript gebouwd worden die visueel beter, sneller en flexibeler werken. Bijvoorbeeld formulieren kunnen slimmer werken en er kunnen visuele effecten gemaakt worden die met alleen HTML en CSS niet of zeer ingewikkeld te maken zouden zijn.

Daarnaast is er vanuit een website vaak behoefte om actuele informatie te versturen of op te halen van het internet. Dit kan door te communiceren met servers waar een database op staat of een bibliotheek met audio en videomateriaal. Dat soort informatie wordt meestal niet compleet naar de browser gestuurd, maar alleen geraadpleegd als het nodig is. Hiervoor zijn vanuit de browser diverse technieken in JavaScript beschikbaar zoals AJAX, de Fetch API en XMLHttpRequest. Wat deze technieken overeenkomstig hebben, is dat ze informatie kunnen ophalen van het internet zonder de browser te verversen.

Wat aan de serverkant nodig is, is een applicatie die antwoord kan geven op deze verzoeken. De keuze voor techniek maakt voor de website geen verschil, het antwoord zal altijd in een bekend formaat zijn zoals HTML, XML of JSON. JavaScript kan hier in de browser weer mee verder werken.

Aan de serverkant zijn de applicaties gebouwd in een taal die past bij de soort dienst (database, AI, streaming) die door de server wordt aangeboden. Voorbeelden zijn Perl, PHP, ASP, JSP (Java) en Ruby. Toch hoort ook JavaScript in dit rijtje thuis.

Server-side applicaties van JavaScript

Server-side JavaScript bevat geen onderdelen die te maken hebben met de browser, maar onderdelen die kunnen communiceren met andere diensten op de server. Met basiskennis JavaScript is deze server-sideversie meestal sneller te leren dan een geheel nieuwe programmeertaal.

Het voordeel van server-side JavaScript is dat deze code alleen bestaat op de server en niet voor gebruikers is in te zien. De resultaten worden in een van de hiervoor genoemde formaten naar de browser verstuurd en er is geen inzage mogelijk in gegevens die de server niet kan of mag sturen. Zo kunnen bijvoorbeeld wachtwoorden of bestandslocaties gebruikt worden zonder dat eindgebruiker die kan zien.

Server-side JavaScript (SSJS) is de beschrijving van een JavaScript-scriptomgeving voor servers. Er zijn diverse applicaties die deze omgeving ondersteunen, waarvan Node.js de bekendste is. Het maakt verbinding met databases mogelijk en ondersteunt veel van de uitgebreide functies van andere scripttalen aan de serverzijde. Het stelt je bijvoorbeeld in staat webpagina's aan databases te koppelen en slaat de clientstatus op, zodat de computer onthoudt waar de client zich bevindt tijdens een proces van meerdere pagina's (zodat cookies minder vaak nodig zijn).

Node.js

Node.js is een JavaScript-runtime-omgeving die JavaScript de mogelijkheid biedt om te worden gebruikt voor server-side scripting. Het vergemakkelijkt het maken van webservers met behulp van modules.

Toepassingen die zijn ontwikkeld met Node.js, kunnen worden uitgevoerd op de meeste servers, ongeacht of die werken met Linux, Windows of andere besturingssystemen.

Een belangrijk voordeel van Node.js is dat het geheel event-driven is. Dat houdt in dat als er geen verzoeken bij de server binnenkomen Node.js in een slaaptoestand verkeert. Dit is energiezuinig en werkt dus goed op kleine servers.

Referenties:

https://nodejs.org/

Introduction to the server side - Learn web development | MDN (mozilla.org)

Express web framework (Node.js/JavaScript) - Learn web development | MDN (mozilla.org)