#TODO “ ” -> ‘ ’

In deze paragraaf je kennismaken met JS. Om JS te kunnen gebruiken heb je niet meer nodig dan een tekstbewerker en een browser. Als je al Atom gebruikt is het ook hier erg handig voor. Als je meer wilt weten dan dat hier behandeld wordt kun je altijd zoeken op <https://www.w3schools.com/js/default.asp>.

Er zijn meerdere manieren om de JS code toe te voegen aan een webpagina. De standard is een vorm van een external include, dit betekent dat het in een apart bestand staat, los van eventuele HTLM. Het liefst zetten we deze external include ook nog op het eind van een pagina. Op die manier kan de gebruiker eerst al de inhoud van de pagina zien en na even te wachten ook de interactie gebruiken. De pagina lijkt dan sneller te zijn, terwijl het eigenlijk even lang zal laden. Een voorbeeld van zo’n soort include is:

#CODE

<html>

<head>

<meta charset="UTF-8" />

</head>

<body>

<header>

</header>

<p>

JS external include

</p>

<footer>

</footer>

<script src="JS\_LOCATIE"></script>

</body>

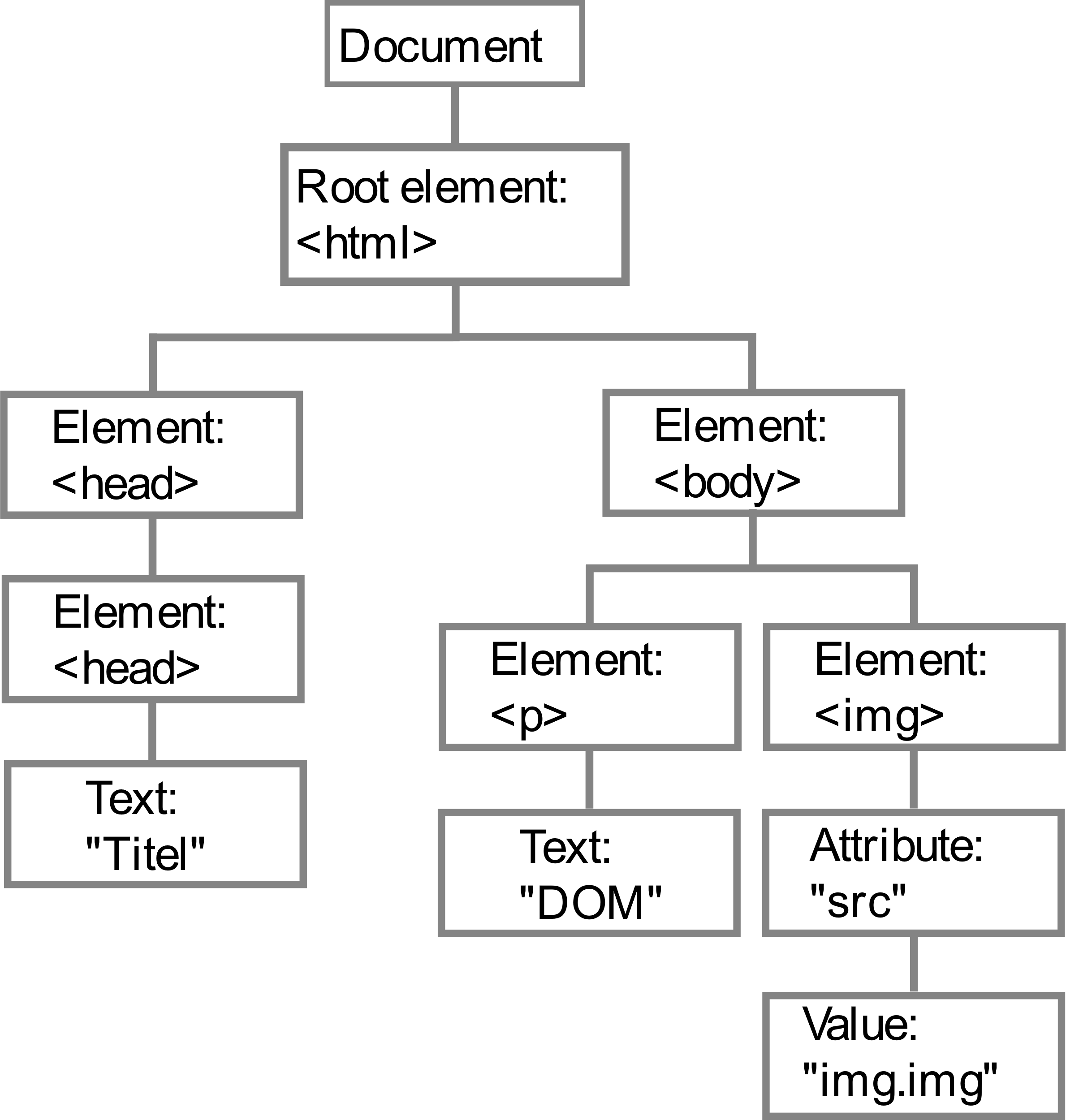
</html>

Je ziet onderdaan #CODE <script src="JS\_LOCATIE"></script> staan, dit is de regel die zorgt voor het inladen van de JS code. Je kunt vervolgens een apart bestand aanmaken dat eindigd met .js en hierin jouw JS gaan schrijven.

Je kunt testen of jouw include werkt door in het JS bestand #CODE alert('test'); te plaatsen. Dit zou moeten zorgen voor een pop-up bericht.

Een van de belangrijkste onderdelen van JS is DOM traversal, dit is eigenlijk het selecteren van HTML onderdelen. DOM staat voor document object model. De DOM is de hiërarchie van de HTML pagina inclusief de attributen van elementen met hun waardes.

Een voorbeeld van een (simpel) DOM is het volgende:



Je ziet dat het document zelf boven staat, met daarin <html>, alle andere elementen vallen daar weer onder. Alles dat in <body> staat zal daar dus ook in de DOM te vinden zijn. Veel elementen hebben ook nog inhoud of attributen en deze zijn ook weer terug te vinden in het DOM.

Met JS ga je gebruik maken van het DOM om elementen te veranderen of informatie eruit te halen. Dit kan op verschillende manieren. De makkelijkste manieren zijn via het ID en de class, deze ken je hopelijk nog van jouw gebruik van CSS.

Om via een ID een element te op te halen gebruik je:

#CODE

document.getElementById(“ID”)

Om het via een class te doen gebruik je:

#CODE

document.getElementByClass(“CLASS”)

Als je het gewenste element hebt opgehaald kun je hier informatie uithalen of aanpassen. Per attribuut van een HTML element vereist dit andere code. Een paar veel gebruikte volgen.

Het ophalen van tekst via:

innerText

Een voorbeeld is:

#CODE

Content = document.getElementById(“formulier”).innerText;

Het toevoegen van een class via:

classList.add(‘CLASS’)

Een voorbeeld is:

#CODE

document.getElementById(“element”).classList.add(‘Class’)

Het verwijderen van een class via:

classList.remove(‘CLASS’)

Een voorbeeld is:

document.getElementById(“element”).classList.remove(‘Class’)

Je ziet dus steeds ELEMENT.METHODE als je met het DOM werkt in JS. Let er ook op dat bij JS op het eind van de regel een ; gebruikt wordt.

#VRAGEN

1. Wat staat in het DOM?
2. Hoe voeg je JS het beste toe aan jouw HTML bestand?
3. Hoe haal je een element met als ID de waarde 3 op met JS?
4. Hoe pas je de inhoud van een <p> met als ID de waarde p aan naar “JS”?
5. Maak een HTML bestand met daarin 3 <p> en stel de inhoud ervan in met JS.

#ANTWOORDEN

1. De hiërarchie van de HTML element, inclusief attributen en hun waardes.
2. Met #CODE <script src=”JS\_LOCATION”></script> net voor </body>.
3. Document.getElementById(‘3’)
4. Document.getElementById(‘3’).innerText = “JS”;
5. Eigen antwoord leerling.