# Section 1 – Fundamentals

Without defer or async in the <header><script> declaration of the javascript(js) associated with a HTML, the browser encounters an error because the js script attempts to modify the HTML body section before it is rendered. The traditional way of addressing this is to simply move the script tag to the bottom of the html file. This introduces problems as there is a strong likelihood that some js should execute on document load initialization and while the document is loading.

The 'async' keyword minimizes render blocking, when the js script is fully-loaded it executes and modifies the html doc.

The 'defer' keyword loads the html doc and js script in-parallel and only executes the js script when the html doc is fully-loaded.

Loading js in the footer is now not an industry-standard, and loading using async or defer in the header is ideal.

When does the browser execute JavaScript?

By default: When the script is encountered. If the script is set to "async", when the script is fully loaded. If the script is set to "defer", when the entire HTML page is rendered.

What is the correct markup for adding an external JavaScript file to an HTML document?  
<script src="javascript.js" async></script>

While `<script src="javascript.js"></script>` is technically correct, it is recommended to always async or defer your script unless you have a specific reason for the script to cause render blocking.

What happens when you defer JavaScript?

The browser loads the JavaScript asynchronously when it is encountered, then waits until all HTML is rendered before executing the script.

JavaScript modules are heavily used in frameworks like React and Vue. What is the advantage of using modules?

Modules enable modularization of code where individual functions, components, data objects, and other parts can be separated into individual files.

Also, type=”module” is automatically-deferred thereby preventing render blocking when you modularized sections / objects / methods of js code.