# How does the browser actually render a website

## Introduction

The topic is all about what happens when we get the HTML and how it comes and prints to our page and we will understand some of the terms of what is happening behind the screen.

Topics to cover:-

* + - **High level view**
    - **In-depth view**
    - **Performance insights**

It is about, how the HTML is parsed, how it gets through the different processes in the browser

## The Browser

It is probably one of the most complex applications that we use it for 90% of our interaction with the computer.

Components that make up a browser:

* Bindings
* Rendering: Parsing, Layout, painting etc.
* Platform
* JavaScript VM

**Rendering: Parsing, layout, painting etc**

**Bindings**

**JavaScript VM**

**Platform**

Bindings is all about binding a lot of operating systems, so when it talks to the network, it will use certain APIs depending on the operating system, Mac or windows. Rendering is about actually constructing the website from the HTML that gets send to it. And there comes the platform, that is dependent on windows or OS x, there are different things between the operating systems, then JavaScript Virtual Machine.

## High level flow:

Paint

Layout

Render Tree

Parse CSS

Parse HTML

In this high level flow, we have two processes, parse HTML & parse CSS, combined to the render or frame tree and then it layout the render tree, and then painting, which is the operation of drawing the graphics and giving us the visual output.

### Parsing

* HTML is forgiving by nature. We can make lots of mistakes and it will still work for us.
* That means the parsing isn’t straight forward. In most languages if we make a mistake or throw an error, it will error out. HTML will try to recover.
* It can be halted as a script can alter the document. Link and style could halt JS execution.
* Its re-entrant.

Script Execution

DOM Tree

Tree Construction

Tokeniser

**<Script> at the bottom**

* Parse uninterrupted
* Faster to render
* Defer and async attributes

## CSS parsing

* Css parsing is pretty standard
* It will create the CSS object model, like the DOM object model,
* We got style sheets, rules, selectors, decorations

## Render/frame tree

DOM +CSSOM

* Combines the two object models, style resolution
* This is the actual representation of what will show on screen

## Calculating visual properties

* Combines all styles
* Default, external, style elements and inline
* Complexity around matching rules for each element
* Style computation

## Recursive process

* Traverse render tree
* Nodes position and size
* Layout its children

## Paint setup

* Will take the layed out render trees
* Creates layers
* Incremental process
* Builds up over 12 phases