**Browser:**

*Binding*

*Renderins:parsing,layout,paitinge,exc….*

*Platform*

*Javascript vm*

***Binding:***

Data binding, or updating the view (usually UI) from your application’s code is probably the holy grail of browser-based web frameworks — the reason they exist, and the reason [web standards have tried to shift](https://developers.google.com/web/fundamentals/getting-started/primers/customelements" \t "https://medium.com/dartlang/_blank) in order to make it easier to “use the platform”.

In this blog post I’ll be reviewing four forms of data binding, and writing a simple library/throw-away framework for each — mostly as a learning experience. I’ll be using [Dart](https://www.dartlang.org/" \t "https://medium.com/dartlang/_blank), which is a high-level programming language I work on at Google (where I work on the AngularDart team), but most of the concepts below apply to JavaScript, TypeScript, or OtherScript flavors.

***Renderin:***

The CSSOM and DOM trees are combined into a render tree, which is then used to compute the layout of each visible element and serves as an input to the paint process that renders the pixels to screen. Optimizing each of these steps is critical to achieving optimal rendering performance.

In the previous section on constructing the object model, we built the DOM and the CSSOM trees based on the HTML and CSS input. However, both of these are independent objects that capture different aspects of the document: one describes the content, and the other describes the style rules that need to be applied to the document. How do we merge the two and get the browser to render pixels on the screen?

### TL;DR

* The DOM and CSSOM trees are combined to form the render tree.
* Render tree contains only the nodes required to render the page.
* Layout computes the exact position and size of each object.
* The last step is paint, which takes in the final render tree and renders the pixels to the screen.

***Platform:***

A platform is a group of technologies that are used as a base upon which other applications, processes or technologies are developed.

In personal computing, a platform is the basic hardware (computer) and software (operating system) on which software applications can be run. This environment constitutes the basic foundation upon which any application or software is supported and/or developed.

Computers use specific central processing units (CPUs) that are designed to run specific machine language code. In order for the computer to run software applications, the applications must be in that CPU’s binary-coded machine language.

Thus, historically, application programs written for one platform would not work on a different platform.

***Javascript vm:***

The JavaScript engine is the component of the browser that takes your JavaScript code, optimizes and executes it.

Also known as virtual machines, JavaScript engines **execute your code in an environment** that is platform-independent.

This means that you can run the same JavaScript code on MacOS, Windows, or Linux.