

# How Do Web Browsers Work



Abenezer Nigos Yakob

If you find any typos, sorry :D

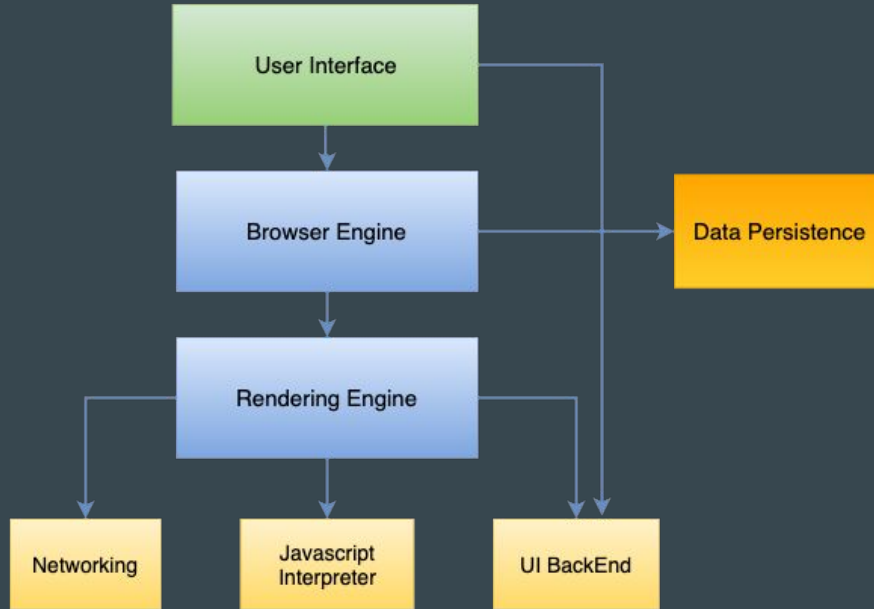
# What is a browser

- A web browser is an application for accessing websites. When a user requests a web page from a particular website, the browser retrieves its files from a web server and then displays the page on the user's screen. Browsers are used on a range of devices, including desktops, laptops, tablets, and smartphones. In 2020, an estimated 4.9 billion people will have used a browser.<sup>[1]</sup> The most used browser is Google Chrome, with a 65% global market share on all devices, followed by Safari with 18%
- Source wikipedia

## How does a browser work?



# What happens behind the scenes ?



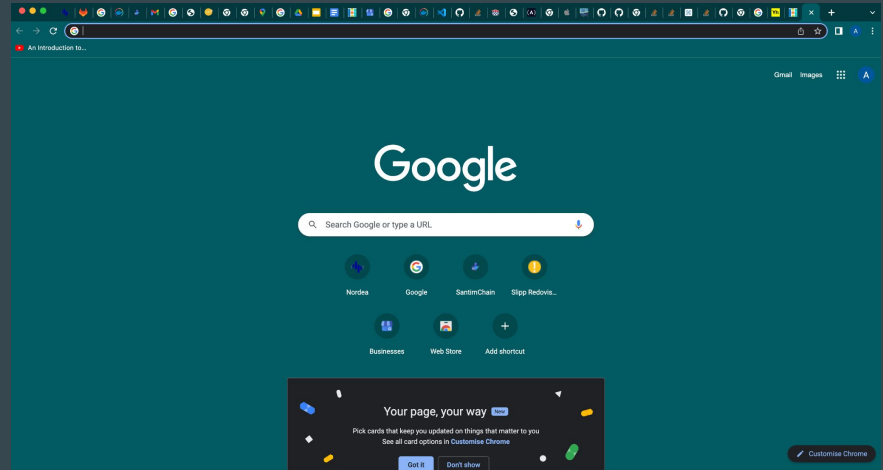
- User interface
- Browser engine
  - data persistence
- Rendering engine
  - Network
  - Javascript interpreter
  - UI backend

# User interface

User interface is everything you see on the browser that you can not manipulate.

## Exemple

- Address bars,
- Settings,
- Back buttons,



# Browser engine

Works as a mediator between the user interface, the rendering engine and the data persistence.

- Owns the browser state

Example of Actions :

- When a person fills in the url and make a request to a web server
- When a person/user clicks on the browser Back button
- Browser Settings
- Dev tools

# Data persistence

The data persistency component is handled by the Browser engine.

- Data storage,
- Local storage,
- Cookie,
- File system

# Rendering engine

Rendering engine parses the HTML, CSS, JS and renders the page based on the files received.

1. **Networking**

Make sure the file we are trying to access are loaded, images, api requests

2. **Javascript interpreter**

The javascript interpreter will parse and execute the javascript in the rendering engine. Chrome uses V8 javascript engine, safari uses Nitro, firefox uses Spider Monkey

3. **UI Backend.** Inputs, select, box



# How the rendering engine works

- **Parsing** : Parses files and creates DOM, CSSOM.
- **Render Tree**: The render tree takes the dom tree and the CSSom tree and creates a structure to be painted on the screen.
- **Layout**: Calculating positions and size on the render tree
- **Paint**: Adds color to the elements on render tree

# Parsing

Parsing : Translating a document into a structure that a code can understand and use

- **Unconventional**
  - Html is a Document type definition (DTD) Creates the DOM
  - The unconventional parser fills in the missing or unclosed html tags and displays valid html tags
- **Conventional**
  - CSS, the conventional parser parses the css file and creates a style tree called CSSOM.
    - Inline css
    - Style tag
    - Style file
  - JS, the javascript is tags is passed to the javascript interpreter
    - Script tag
    - Script file

# Render Tree

Render tree is generated while Dom tree is constructed, the render tree is basically the combination of the dom and the CSSom

- Render tree handels visual elements and in which order they need to be displayed, the render tree have Switch cases if the DOM element need to be displayed and how
  1. Render none
  2. Render inline
  3. Render block
  4. Render inline-block
  5. Render list-item

# Layout

The process of calculating positions and size is done in the layout flow, the layout process is a recursive process that begins at the `<html>`

- global layout: the resize of browser or changing font can recalculate the entire layout
- Incremental Layout: using to recalculate the layout of a specific dom item using the dirty-bit System:

# Paint

paint method goes through the render tree and adds color to the elements.

- global painting
- incremental painting using the same dirty-byte-system as layout to paint specific elements in the render tree.

When painting is done process is done the browser engine will mark the document ready, so the user can interact with it

- painting order

1. Background color
2. Background image
3. Border
4. Children
5. Outline