Realtor

0. Why Realtor?

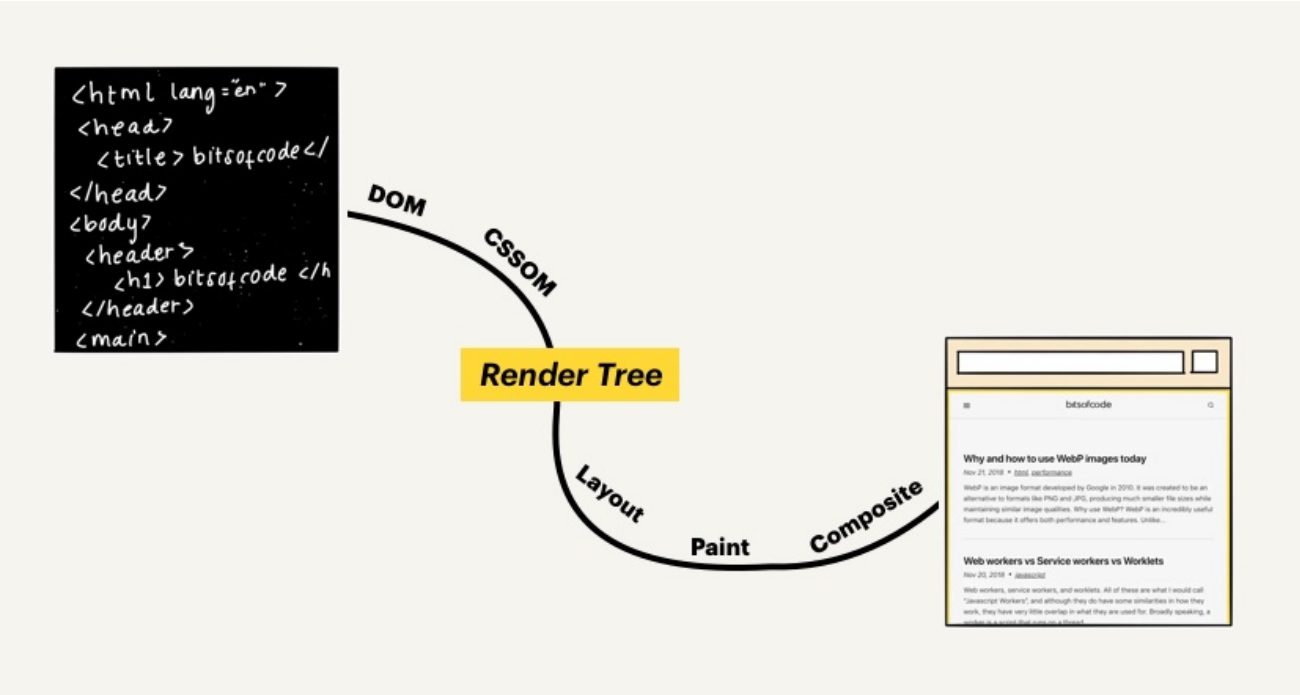
Realtor a leader in online real estate. It helps people find the right home for them. My friend Angela told me that Realtor is a place to learn a lot, make an impact, grow my career. Why I saw this Full Stack Web Developer position, I was interested since it is a unique opportunity to work on back-end and front-end.

1. What is the DOM(Document Object Model)?

A web page built can be grouped into two stages.
1. The first stage involves the browser parsing the document to determine what will ultimately be rendered on the page.
2. The second stage involves the browser performing the render.

The result of the first stage is what is called a **"render tree"**. The render tree is a **representation of the HTML elements** that will be **rendered on the page** and **their related styles**. In order to build this tree, the browser needs two things:

1. The **CSSOM**, a representation of the styles associated with elements
2. The **DOM**, a representation of the elements



The DOM is an object-based representation essentially an attempt to **convert the**

structure and content of the HTML document into an **object model** that can be used by various programs.

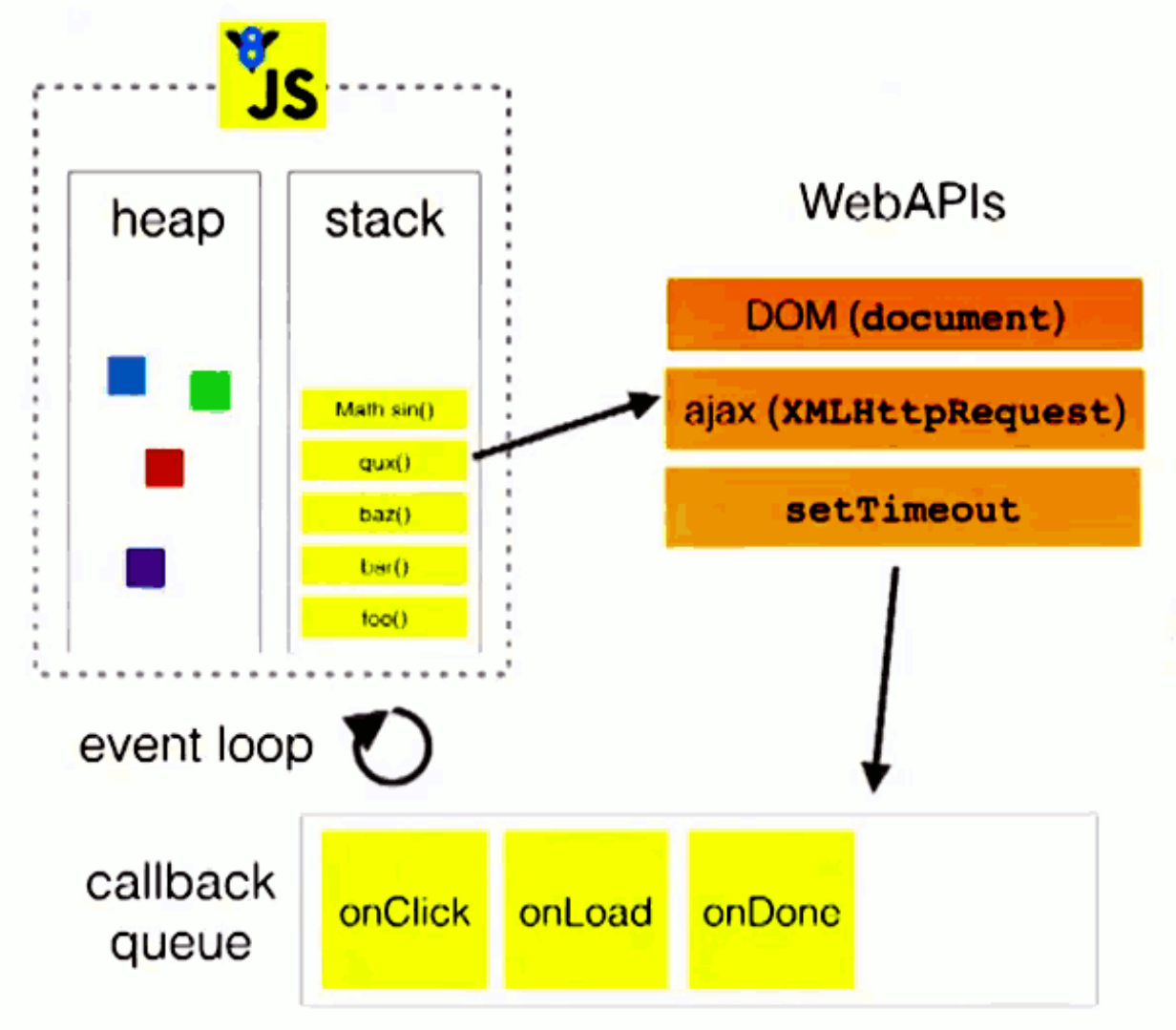
The object structure of the DOM is represented by what is called a **“node tree”**. It is so called because it can be thought of as a tree with a single parent stem that branches out into several child branches, each which may have leaves.

2. What is the event loop? (Single-threaded, non-blocking, asynchronous, concurrent, runtime)

JavaScript has a **concurrency model(并发模型)** based on an **event loop**, which is responsible for **executing the code, collecting and processing events, and executing queued sub-tasks**.

1. **同步任务（synchronous）**在主线程上排队执行的任务，同步任务指的是，只有前一个任务执行完毕，才能执行后一个任务。
2. **异步任务（asynchronous）**异步任务指的是，不进入主线程、而进入"任务队列"（task queue）的任务。

只有"任务队列"通知主线程，某个异步任务可以执行了，该任务才会进入主线程执行。



http://www.ruanyifeng.com/blog/2014/10/event-loop.html
https://vimeo.com/96425312

3. What is a closure?

A **closure** is a **function** having access to the **parent scope**, even after the **parent function has closed**.

Classical Inheritance	Prototypal Inheritance
Classes are immutable. You can't modify or add new methods to them at runtime.	Prototypes are flexible. They may be either mutable or immutable.
Classes may or may not support multiple inheritance.	Objects can inherit from multiple prototypes.
It's verbose and complicated. You have abstract classes, final classes, interfaces, etc.	It's simple. You only have objects and extending objects is the only operation required.

4. What is this?

The **this** keyword evaluates to the value of the ThisBinding of the **current execution context**. In short, The JavaScript **this** keyword refers to the object it belongs to.

1. In a method, **this** refers to **the owner object**.
2. Alone, this refers to **the global object**.
3. In a function, this refers to the **global object**.
4. In a function, in strict mode, **this is undefined**.
5. In an event, this refers to **the element that received the event**.
6. Methods like **call(), and apply()** can refer **this** to **any object**.

5. What is event bubbling and how does it work?

Event bubbling **directs an event** to its **intended target**, it works like this:

1. A button is clicked and the event is directed to the button
2. If an event handler is set for that object, the event is triggered
3. If no event handler is set for that object, the event bubbles up (like a bubble in water) to the objects parent

The event bubbles up from parent to parent until it is handled, or until it reaches the document object.

Event bubbling and **capturing** are two ways of **event propagation** in the HTML

DOM API, when an event occurs in an element inside another element, and both elements have registered a handle for that event.

1. With bubbling, the event is **first captured and handled by the innermost element** and then propagated to outer elements.
2. With capturing, the event is **first captured by the outermost element** and propagated to the inner elements.

6. What is REST, and why do people use it?

REST (**Representational State Transfer**) is an architectural style. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations.

REST defines some 'verbs' in order to interact with the resources. Some of these are:

1. GET: to receive the resource representation
2. POST: to add some information to the resource
3. PUT: modify the resources
4. DELETE: delete the resources