

Module 38-How JavaScript and Browser works

আজকের মডিউল:

জাস্ট ৯টা ভিডিও। প্রথমেই ইন্টারনেট নিয়ে চিরপরিচিত আলোচনা। জাভাস্ক্রিপ্ট এবং ব্রাউজার নিয়ে হালকা মেজাজে আলোচনা। (কারণ ভিডিও এবং ক্যামেরা কিছু ঝামেলা করেছে)। তারপরে ছোট করে setTimeout এবং setInterval এর ভিতরের খুঁটিনাটি অনেক জিনিস জানবে। তারপরে এই রকম দরকারি কিছু জিনিস আলোচনা করা হবে এই মডিউলে। যাতে তুমি জাভাস্ক্রিপ্ট কিভাবে synchronous এবং asynchronous হিসেবে কাজ করে। সেটা বুঝার জন্য লাগবে।

.

আজকে যে যে জিনিসগুলো একটু খেয়াল করতে হবে

১. ইন্টারনেট কিভাবে কাজ করে এইটা না জানলে একটু ধারণা নিয়ে নাও
২. ব্রাউজার কিভাবে কাজ করে। কিভাবে DOM tree, Render Tree বানায় সেটা ছোট করে হলেও নামগুলো জানতে হবে
৩. V8 ইঞ্জিন যে ক্রোম এর ভিতরে জাভাস্ক্রিপ্ট কোড রান করে আউটপুট দেয় সেটা জানতে হবে। আরেকটু বেশি জানতে চাইলে JIT সম্পর্কে নোটস নিয়ে রাখতে পারো
৪. setTimeout কি জিনিস। এইটা কিভাবে কাজ করে সেটা একটু মাথায় থাকতে হবে
৫. event loop লুপ কি জিনিস। এই রিলেটেড পুরা জিনিসটা আজকে অনেকেই বুঝবে না। তবে দেখে রাখা লাগবে। নামগুলো নোটস নিয়ে রাখবে।

38-0 Milestone overview

38-1 Module Introduction and how internet works

The **Domain Name System** (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

There are four layers of the TCP/IP model: **network access, internet, transport, and application**. Used together, these layers are a suite of protocols. The TCP/IP model

passes data through these layers in a particular order when a user sends information, and then again in reverse order when the data is received.

38-2 How Browser works, DOM tree, Render Tree

Browser হল এক ধরনের desktop software যা PC তে install করে ব্যবহার করতে হয়। কোনো একটি browser এ static website দেখানোর steps :

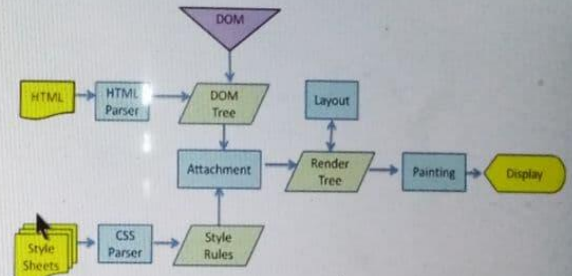
Browser আমাদের html file টা কে একটা tree তে পরিবর্তন করে যা **DOM tree** হিসেবে পরিচিত।

html file এর উপর style দেওয়ার জন্য browser একটা style tree তৈরী করে।


DOM tree এবং **style tree** মিলে তৈরী হয় **render tree** যা website এর layout কে নির্দেশ করে। Dom tree/render tree তে কোনো কিছু update হলে সেই অনুযায়ী **re-rendering** হয়।

Render tree পাওয়ার পর সেটার উপর **paint** করা হয়। **Re-render** হলে **re-paint (reflow)** হয়।

সবশেষে **User Interface (UI)** এ output টা **display** হয়।



```
graph LR
    HTML[HTML] --> HTML_Parser[HTML Parser]
    HTML_Parser --> DOM_Tree[DOM Tree]
    Style_Sheets[Style Sheets] --> CSS_Parser[CSS Parser]
    CSS_Parser --> Style_Rules[Style Rules]
    DOM_Tree --> Attachment[Attachment]
    Style_Rules --> Attachment
    Attachment --> Render_Tree[Render Tree]
    Render_Tree --> Layout[Layout]
    Layout --> Render_Tree
    Render_Tree --> Painting[Painting]
    Painting --> Display[Display]
```



38-3 JavaScript Engine V8 Internal mechanism

V8 is Google's open source high-performance JavaScript and WebAssembly engine, written in C++. It is used in Chrome and in Node.js, among others. It implements [ECMAScript](#) and [WebAssembly](#), and runs on Windows 7 or later, macOS 10.12+, and Linux systems that use x64, IA-32, ARM, or MIPS processors. V8 can run standalone, or can be embedded into any C++ application.

38-4 setTimeout simple Asynchronous JS using

The `setTimeout()` method **executes a block of code after the specified time**. The method executes the code only once. The commonly used syntax of JavaScript `setTimeout` is: `setTimeout(function, milliseconds);`

```
function doSomething()
{
```

```

    console.log("I am coding JavaScript");
}

console.log('First: ami sobar age');
console.log('Second: I am the second person');
// doSomething();
setTimeout(doSomething);

setTimeout(doSomething,5000);//(5000->milisecond)

setTimeout(function(){
    console.log("I am using vs code");
},500)

setTimeout(() => {
    console.log("Exploring MDN articles");
}, 4000);

console.log('Third: ami 3 number baccha');
console.log('Fourth: ami ek hali purno korsini');

```

38-5 Recognize fetch as an Asynchronous activity

What is asynchronous JavaScript?

Async operations like promises are put into an event queue, which runs after the main thread has finished processing so that they do not block subsequent JavaScript code from running. The queued operations will complete as soon as possible then return their results to the JavaScript environment.

What is synchronous JavaScript?

Synchronous JavaScript: As the name suggests **synchronous means to be in a sequence**, i.e. every statement of the code gets executed one by one. So, basically a statement has to wait for the earlier statement to get executed. Let us understand this with the help of an example.

```

console.log(111111);
console.log(222222);
setTimeout(() => console.log('aaaaaaa'), 5000);
fetch("https://jsonplaceholder.typicode.com/todos/1")
    .then((response) => response.json())
    .then((json) => console.log(json));

```

```

console.log(333333);
console.log(444444);
console.log(77777);

for(let i=0;i<1000;i++){
    console.log(i);
}

```

38-6 setInterval and clearInterval with x++ and ++x

What is setInterval in JavaScript?

setInterval() The setInterval() method, offered on the Window and Worker interfaces, **repeatedly calls a function or executes a code snippet, with a fixed time delay between each call**. This method returns an interval ID which uniquely identifies the interval, so you can remove it later by calling clearInterval()

What does clearInterval do JavaScript?

The clearInterval() function in javascript **clears the interval which has been set by setInterval()** function before that. setInterval() function takes two parameters. ... The number id value returned by setInterval() function is stored in a variable and it's passed into the clearInterval() function to clear the interval

```

console.log("First");

// setInterval(() => {
//     console.log('tik tik tik tik')
// }, 1000)

console.log("Second");

let seconds = 0;

const timeId = setInterval(() => {
    // seconds++;
    console.log(++seconds);
    if (seconds > 15) {
        clearInterval(timeId);
    }
}, 1000)

```

38-7 (advanced) JavaScript event loop and concurrency

38-8 Integrate chrome devtool console with VS Code

38-9 Module summary and HTTP layers