*“The virtual DOM is an in-memory representation of the real DOM elements generated by React components before any changes are made to the page.”.*

virtual DOM like a blueprint. It contains all the details needed to construct the DOM, but because it doesn't require all the heavyweight parts that go into a real DOM, it can be created and changed much more easily

Ordinary JS traverses or renders the whole DOM instead of rendering only the part which requires changes.

So whenever you have any changes, as in you want to add another <div> to your DOM then the virtual DOM will be created which actually does not do any changes in the actual DOM. Now with this virtual DOM, you will be checking the difference between this and your current DOM. And only the part which is different (in this case the new <div>) will be added instead of re-rendering the whole DOM.

 React is using a fake DOM and not a real one which is Virtual DOM

Virtual DOM was not invented by React, but React uses it and provides it for free

This is what allows react to be fast. It does this by means of virtual DOM diffing. Comparing two virtual trees — old and new — and make only the necessary changes into the real DOM.

VDOM is a strategically to update DOM without redrawing all the nodes in a single page application. Finding a node in tress structure is easy but DOM tree for an SPA app can be drastically huge. Finding and updating a node/nodes in case of an event is not time efficient.

VDOM solve this problem by creating a high label abstraction of actual dom. **The VDOM is a high level lightweight in-memory tree representation of actual DOM.**

For example, consider adding a node in DOM; react keep a copy of VDOM in memory

1. Create a VDOM with a new state
2. Compare it with older VDOM using diffing.
3. Update only differ nodes in real DOM.
4. Assign new VDOM as older VDOM.

========================================

features from ES6, a recent version of JavaScript. In this tutorial, we’re using [arrow functions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions), [classes](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Classes), [let](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/let), and [const](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/const) statements.

Arrow functions

**var materials = [**

**'Hydrogen',**

**'Helium',**

**'Lithium',**

**'Beryllium'**

**];**

**console.log(materials.map(material => material.length));**

// expected output: Array [8, 6, 7, 9]