ECAP472

Web Technologies

Dr. Pritpal Singh

Associate Professor

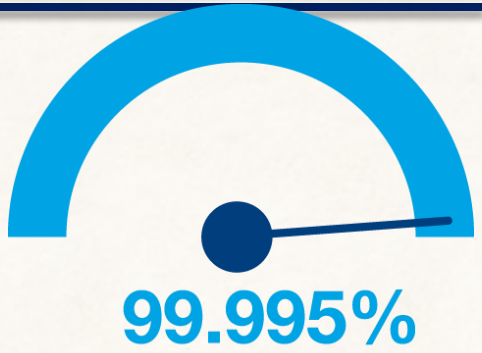
Learning Outcomes

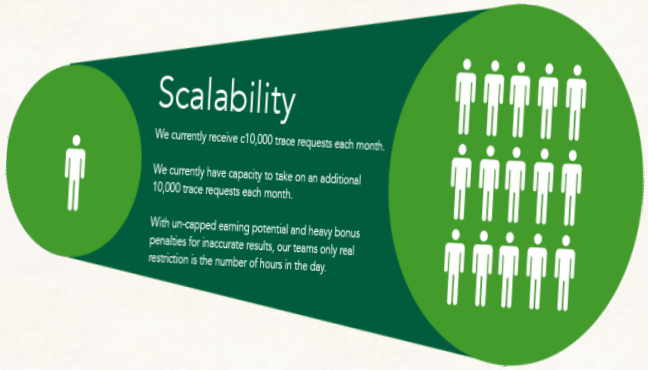
After this lecture, you will be able to

- Understand concept of React Dom, The Virtual Dom and React Elements.

Principles of Web Design

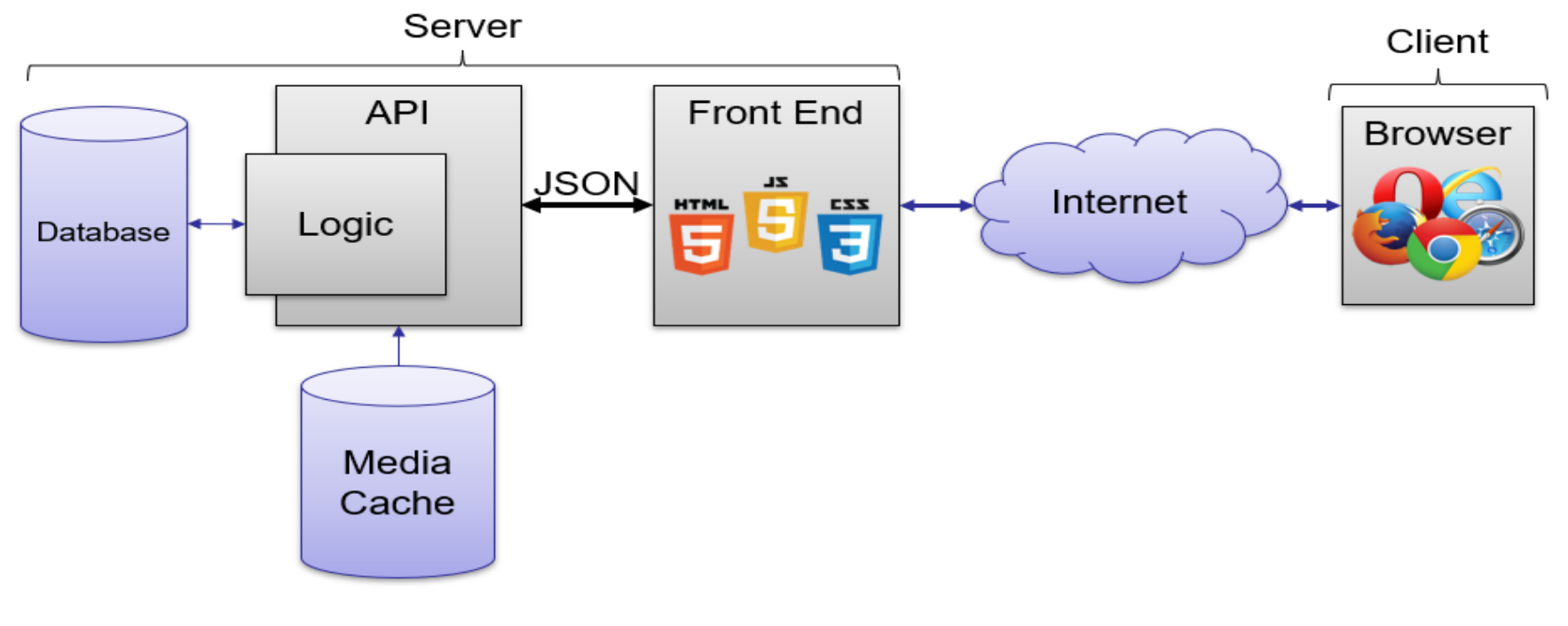
- Availability
- Performance
- Reliability
- Scalability
- Manageability
- Cost





Core Components of Web Applications

- UI (Front End (DOM, Framework))
- Request Layer (Web API)
- Back End (Database, Logic)



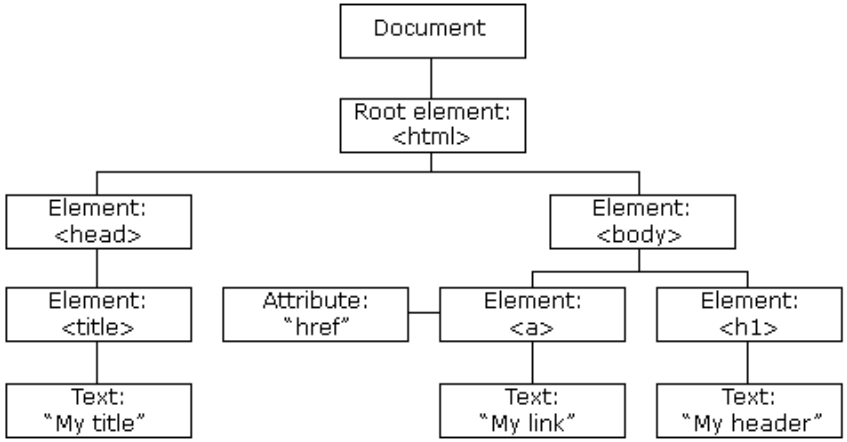
Front End Languages

- HTML/CSS
- JavaScript
- Java (applets)
- What is the most popular?

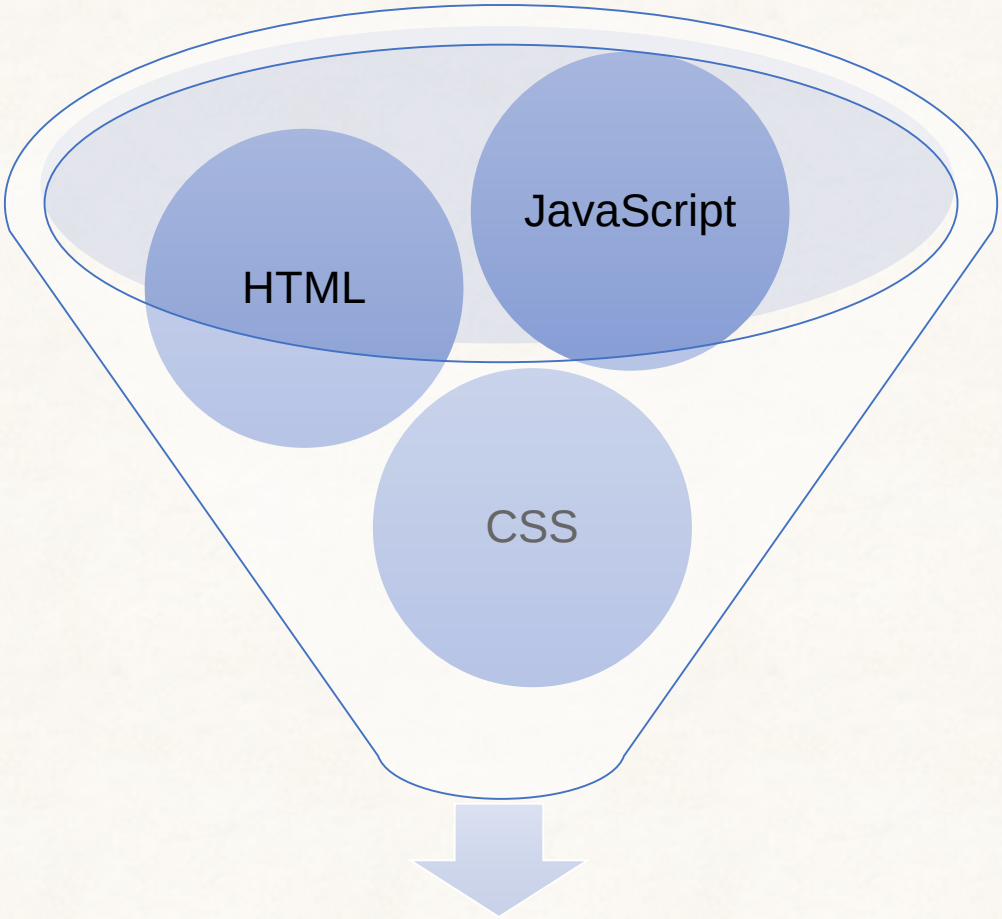
  - Answer: JavaScript/HTML/CSS is the only real option for front-end native languages and is basically the standard. But there are many variations on JavaScript that are used.

DOM (Document Object Model)

- Document Object Model makes every addressable item in a web application an Object that can be manipulated for color, transparency, position, sound and behaviors.
- Every HTML Tag is a DOM object



DOM (Document Object Model)



Document Object Model

According to W3C - "The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

Document Object Model

- The document object represents the whole html document.
- When html document is loaded in the browser, it becomes a document object. It is the root element that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our web page.

Why DOM is required?

- HTML is used to structure the web pages and Javascript is used to add behavior to our web pages. When an HTML file is loaded into the browser, the javascript can not understand the HTML document directly. So, a corresponding document is created(DOM). DOM is basically the representation of the same HTML document but in a different format with the use of objects.

Why DOM is required?

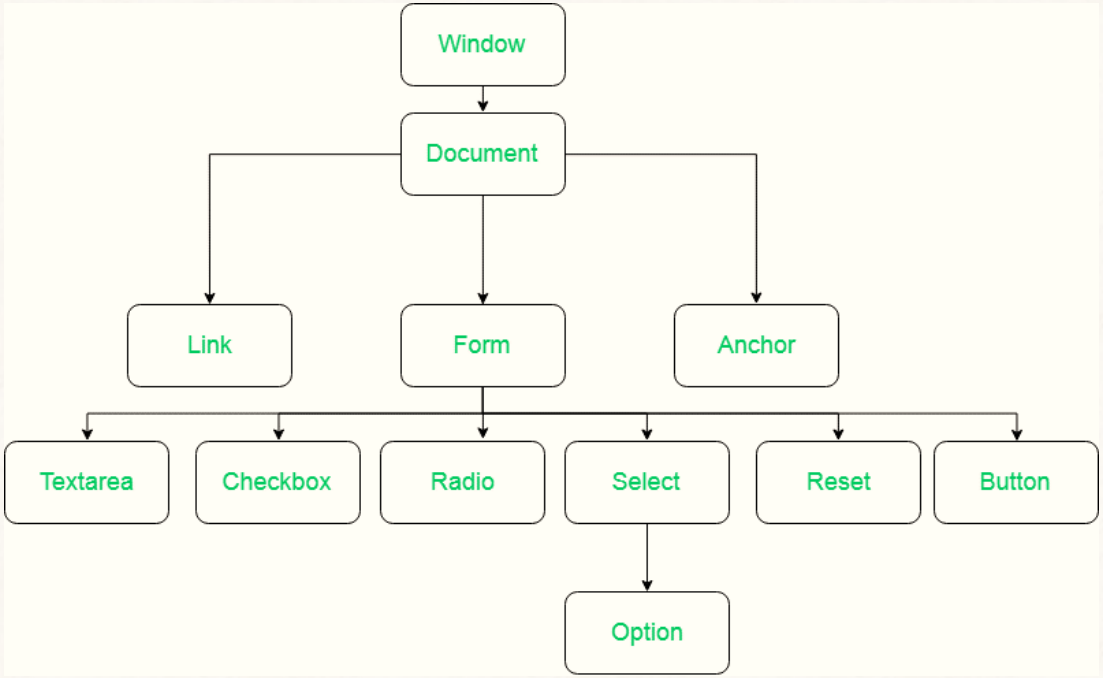
- Javascript interprets DOM easily i.e javascript can not understand the tags(<h1>H</h1>) in HTML document but can understand object h1 in DOM. Now, Javascript can access each of the objects (h1, p, etc) by using different functions.

Why called an Object Model?

- Documents are modeled using objects, and the model includes not only the structure of a document but also the behavior of a document and the objects of which it is composed of like tag elements with attributes in HTML.

Properties of DOM:

- Let's see the properties of the document object that can be accessed and modified by the document object.



Properties of DOM:

- Window Object: Window Object is always at top of the hierarchy.
- Document object: When an HTML document is loaded into a window, it becomes a document object.
- Form Object: It is represented by form tags.
- Link Object: It is represented by link tags.
- Anchor Object: It is represented by a href tags.
- Form Control Elements:: Form can have many control elements such as text fields, buttons, radio buttons, and checkboxes, etc.

What is a Framework?

- Software Framework designed to reduce overhead in web development
- Types of Framework Architectures
  - Model-View-Controller (MVC)
  - Push vs Pull Based

What is a Framework?

Most MVC Frameworks user push-based architecture “action based” (Django, Ruby on Rails, Symfony, Stripes)

Pull-based or “component based” (Lift, Angular2, React)

Three Tier Organization

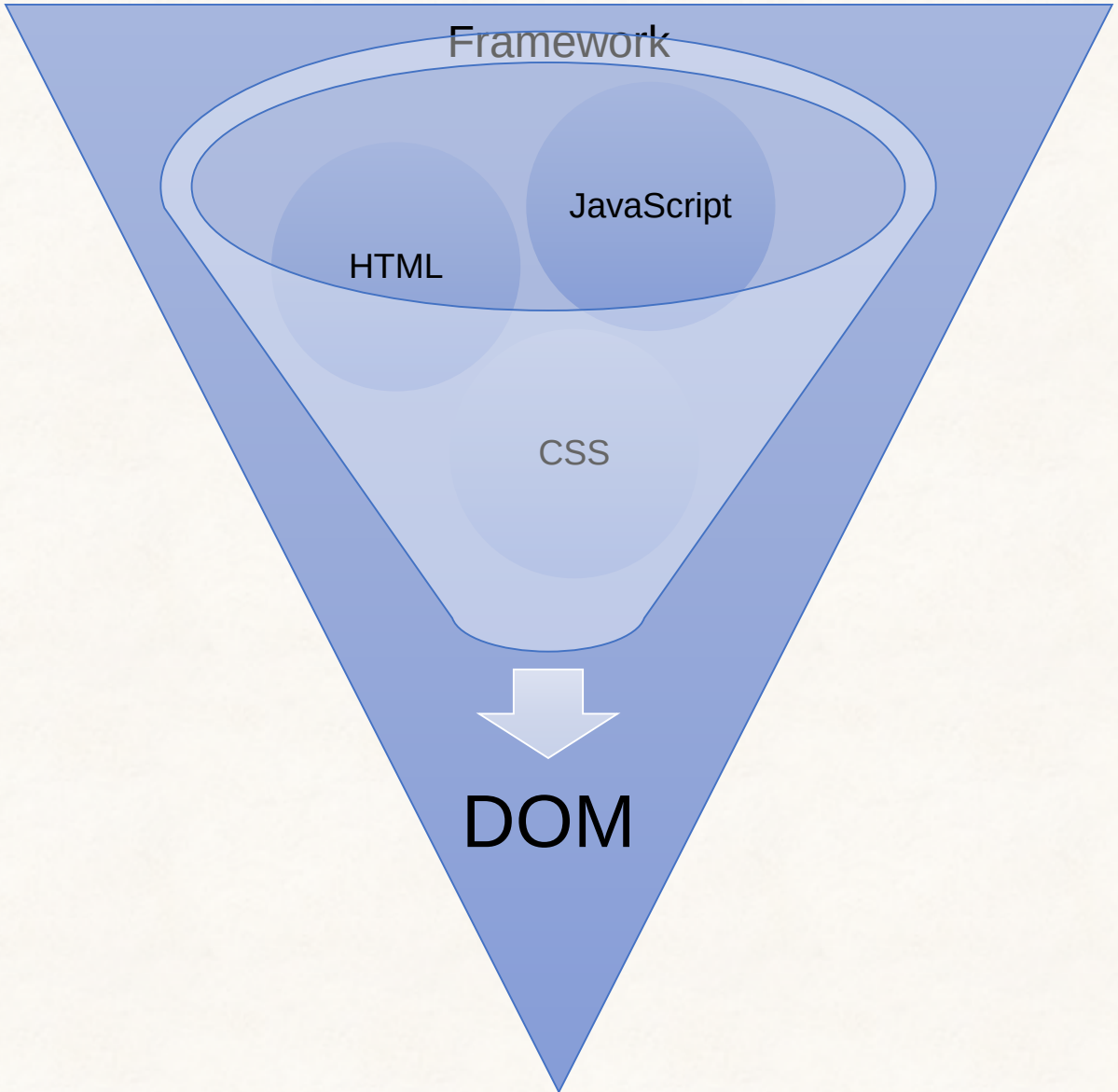
- Client (Usually the browser running HTML/Javascipt/CSS)
- Application (Running the Business Logic)
- Database (Data Storage)

What is a Framework?

Types of Frameworks

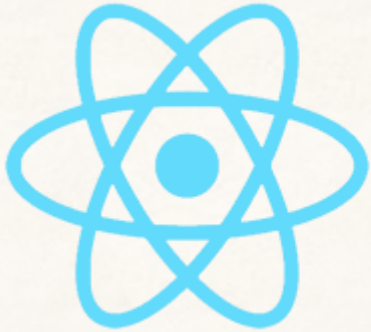
- Server Side: Django, Ruby on Rails
- Client Side: Angular, React, Vue

Framework



Javascript Frameworks

- AngularJS/Angular 2
- ASP.net
- React
- Polymer 1.0
- Ember.js
- Vue.js



React

ember

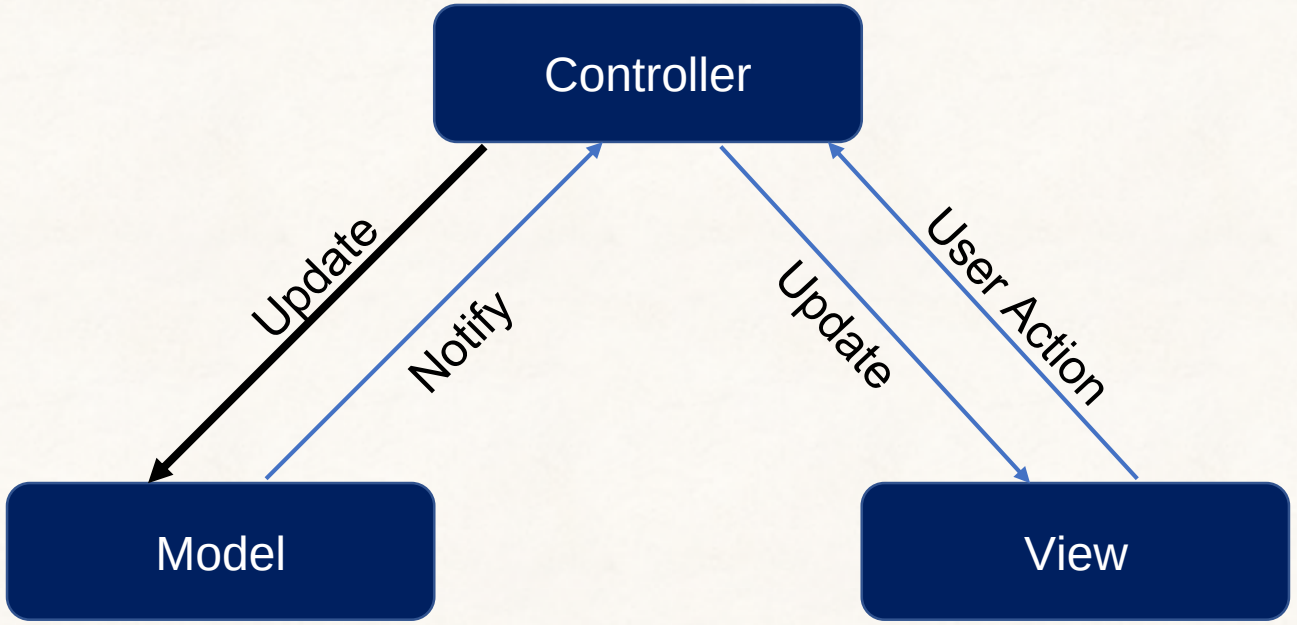
MVC (Model View Controller)

- A Web Application Development Framework
- Model (M):
  - Where the data for the DOM is stored and handled)
  - This is where the backend connects

MVC (Model View Controller)

- View (V):
  - Think of this like a Page which is a single DOM
  - Where changes to the page are rendered and displayed
- Control (C):
  - This handles user input and interactions
    - Buttons
    - Forms
    - General Interface

MVC Model



What is a Backend?

- All of the awesome that runs your application.
- Web API
- Connection layer between the frontend and backend
- Connected through API calls (POST, GET, PUT, etc.)
- Transmit Content from the Backend to the Frontend commonly in JSON Blobs
- Service Architecture that drives everything (Where all the logic is)

What is a WebAPI?

- The intermediate layer between front end and back-end systems
- A “must have” if your APIs will be consumed by third-party services
- Attention to details:
- How consumable is the API (signature, content negotiation)?

What is a WebAPI?

- Does it comply with standards (response codes, etc.)?
- Is it secure?
- How do you handle multiple versions?
- Is it truly RESTful?

Virtual dom

- The virtual DOM (VDOM) is a programming concept where an ideal, or “virtual”, representation of a UI is kept in memory and synced with the “real” DOM by a library such as ReactDOM. This process is called reconciliation

Virtual DOM

- React uses Virtual DOM exists which is like a lightweight copy of the actual DOM(a virtual representation of the DOM). So for every object that exists in the original DOM, there is an object for that in React Virtual DOM.
- It is exactly the same, but it does not have the power to directly change the layout of the document. Manipulating DOM is slow, but manipulating Virtual DOM is fast as nothing gets drawn on the screen.

Virtual DOM

- So each time there is a change in the state of our application, virtual DOM gets updated first instead of the real DOM. You may still wonder, “Aren’t we doing the same thing again and doubling our work? How can this be faster?” Read below to understand how things will be faster using virtual DOM.

How Virtual DOM actually make the things faster:

- When anything new is added to the application, a virtual DOM is created and it is represented as a tree. Each element in the application is a node in this tree. So, whenever there is a change in state of any element, a new Virtual DOM tree is created.
- This new Virtual DOM tree is then compared with the previous Virtual DOM tree and make a note of the changes. After this, it finds the best possible ways to make these changes to the real DOM. Now only the updated elements will get rendered on the page again.

How Virtual DOM helps React

- In react, everything is treated as a component be it a functional component or class component. A component can contain a state. Each time we change something in our JSX file or let’s put it in simple terms, whenever the state of any component is changed react updates it’s Virtual DOM tree.

How Virtual DOM helps React

- Though it may sound that it is ineffective but the cost is not much significant as updating the virtual DOM doesn’t take much time. React maintains two Virtual DOM at each time, one contains the updated Virtual DOM and one which is just the pre-update version of this updated Virtual DOM.

How Virtual DOM helps React

- It just mean that the changes to the real DOM are sent in batches instead of sending any update for a single change in the state of a component.
- We have seen that the re-rendering of the UI is the most expensive part and React manages to do this most efficiently by ensuring that the Real DOM receives batch updates to re-render the UI. This entire proces of transforming changes to the real DOM is called Reconciliation

React Element

- Elements are the smallest building blocks of React apps. An element specifies what should be there in our UI. An Element is a plain object describing what we want to appear in terms of the DOM nodes.

React Element

- Creating a React Element is Cheap compared to DOM elements. An Element can be Created by using JSX or React without JSX
- const element = <h1>Hello, world</h1>;
- Object of type h1 and assigned it to a variable called as element. This element should be rendered into the Browser DOM, and for that we need a container

React Element

- React Element - It is a simple object that describes a DOM node and its attributes or properties you can say. It is an immutable description object and you can not apply any methods on it.
- React Component - It is a function or class that accepts an input and returns a React element.

That’s all for now…