## What are Semantic Elements?

A semantic element clearly describes its meaning to both the browser and the developer.

* <article>
* <aside>
* <details>
* <figcaption>
* <figure>
* <footer>
* <header>
* <main>
* <mark>
* <nav>
* <section>
* <summary>
* <time>

## Meta tag?

The <meta> tag defines metadata about an HTML document. Metadata is data (information) about data.

<meta> tags always go inside the <head> element, and are typically used to specify character set, page description, keywords, author of the document, and viewport settings.

Metadata will not be displayed on the page, but is machine parsable.

Metadata is used by browsers (how to display content or reload page), search engines (keywords), and other web services.

There is a method to let web designers take control over the viewport (the user's visible area of a web page), through the <meta> tag (See "Setting The Viewport" example below).

What are Pseudo-classes?

A pseudo-class is used to define a special state of an element.

For example, it can be used to:

* Style an element when a user mouses over it
* Style visited and unvisited links differently
* Style an element when it gets focus

/\* unvisited link \*/  
a:link {  
  color: #FF0000;  
}  
  
/\* visited link \*/  
a:visited {  
  color: #00FF00;  
}  
  
/\* mouse over link \*/  
a:hover {  
  color: #FF00FF;  
}  
  
/\* selected link \*/  
a:active {  
  color: #0000FF;  
}

**What is boz-sizing**

The box-sizing property allows us to include the padding and border in an element's total width and height.

**What are Pseudo-Elements?**

A CSS pseudo-element is used to style specified parts of an element.

For example, it can be used to:

* Style the first letter, or line, of an element
* Insert content before, or after, the content of an element

Ex

p::first-letter {  
  color: #ff0000;  
  font-size: xx-large;  
}  
  
p::first-line {  
  color: #0000ff;  
  font-variant: small-caps;  
}

after and before also psudo elements

**How to improve react application performance**

## Use React.Fragment to Avoid Adding Extra Nodes to the DOM

## Use Production Build

## Use React.Suspense and React.Lazy for Lazy Loading Components

## Use React.memo for Component Memoization

## Virtualize a Large List Using react-window

**What is the difference between ReactJS and normal JavaScript?**

React is a JavaScript library that allows developers to create single-page apps. React offers a component-based approach to building apps which leads to high reusability of code. **Plain JavaScript library is more for small applications and it offers the ability to build web pages like you would in any web browser**

**What is JavaScript DOM used for?**

The Document Object Model (DOM) is **a programming interface for web documents**. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.

**How do I know if an app is a React?**

Quite Simple. Just **install the developer tools for corresponding frameworks**. For Angular, it is augury and for React, it is React Developer Tools. So add their respective chrome extensions and you are good to know.

**Current version of react**

**18.2.0** / 14 June 2022

**Why is React 18 better?**

**React 18 introduces automatic batching which allows all state updates – even within promises, setTimeouts, and event callbacks – to be batched**. This significantly reduces the work that React has to do in the background. React will wait for a micro-task to finish before re-rendering

## Suspense On the Server

## Automatic Batching in React 18

## Automatic batching means that React will now batch updates you make inside your components. Batching prevents unnecessary renders of your component.

## Transitions

## Urgent updates.

## Transition updates

## 5 New Hooks in React 18

### **useId**

### **useTransition**

### **useDeferredValue**

### **useSyncExternalStore**

### **useInsertionEffect**

**Es7, Es8, Es9, and Es10 Features:**

* string. padEnd() and string. padStart() functions:
* array. prototype. includes()
* Async Functions.
* object. getOwnPropertyDescriptors()
* RegEx changes.
* Rest/Spread properties.
* Asynchronous iteration.
* Array. flat()

**How do I combine multiple reducers?**

You define multiple reducers to handle different pieces of your application's state, then compose these reducers together into one root reducer. The root reducer is then passed into the Redux createStore() method. In order to let us combine multiple reducers together, **Redux provides the combineReducers() method**.

**.** **React.** **js - Client Side Encryption**

1. Step 1: Create the react app:
2. Step 2: Go into your folder and install our dependencies.
3. Step 3: Next you'll need a private and public keys. ...
4. Step 1: Create your Golang app (change according to your configurations):

**How do I make my React site secure?**

1. Use default XSS protection with data binding.
2. Watch out for dangerous URLs and URL-based script injection.
3. Sanitize and render HTML.
4. Avoid direct DOM access.
5. Secure React server-side rendering.
6. Check for known vulnerabilities in dependencies.
7. Avoid JSON injection attacks.

## Refs

Refs is **the shorthand used for references in React**. It is similar to keys in React. It is an attribute which makes it possible to store a reference to particular DOM nodes or React elements. It provides a way to access React DOM nodes or React elements and how to interact with it.

## Reconciliation

When a component’s props or state change, React decides whether an actual DOM update is necessary by comparing the newly returned element with the previously rendered one. When they are not equal, React will update the DOM. This process is called “reconciliation”.

**LESS:**It is a **Leaner Style Sheet**that is dynamic in nature and efficiently enables customization and reusability. LESS supports cross-browser friendly. It is JavaScript-based and has very precise error reporting along with indicating the exact location of the error. It helps in readability and reusability by letting users create properties like variables and mixins to create dynamic and reusable values throughout the project. It uses Preboot.less to implement mixins.

**SASS:**It is a **Syntactically Awesome Style** **Sheet** that supports all version compatible extensions to CSS that increases code reusability. It is implemented using Ruby and actively reports errors made in syntax. It uses compass extension to implement mixins and also enables a user to implement their own functions. It lets users create code reusabilities like variables and mixins as well.

**What is a web worker in JavaScript?**

A web worker is **a JavaScript that runs in the background, independently of other scripts, without affecting the performance of the page**. You can continue to do whatever you want: clicking, selecting things, etc., while the web worker runs in the background

**What are service workers in JavaScript?**

Service workers are **specialized JavaScript assets that act as proxies between web browsers and web servers**. They aim to improve reliability by providing offline access, as well as boost page performance

**Guidelines to speed up your website?**

1. Use a Content Delivery Network (CDN) ...
2. Move your website to a better host. ...
3. Optimize the size of images on your website. ...
4. Reduce the number of plugins. ...
5. Minimize the number of JavaScript and CSS files. ...
6. Use website caching. ...
7. Implement Gzip Compression. ...
8. Database optimization in CMS.

**What is cors?**

Cross-origin resource sharing (CORS) is **a browser security feature that restricts cross-origin HTTP requests that are initiated from scripts running in the browser**. If your REST API's resources receive non-simple cross-origin HTTP requests, you need to enable CORS support.

## Security measures

### Use strong content security policy

### Enable XSS protection mode(xss= Cross-site Scripting)

### Use UI frameworks

### Disable iframe embedding to prevent clickjacking attacks

### Think twice before adding third-party services

### Use Sub-resource Integrity for third-party scripts

### Keep your dependencies up to date

### Don’t set innerHTML value based on the user input

### Don’t leak referrer value

### Limit access to browser features & APIs

## Library and FrameWork

Libraries provide developers with predefined functions and classes to make their work easier and boost the development process. Framework, on the other hand, is like the foundation upon which developers build applications for specific platforms.

**What is call stack**

A call stack is **a mechanism for an interpreter (like the JavaScript interpreter in a web browser) to keep track of its place in a script that calls multiple functions** — what function is currently being run and what functions are called from within that function, etc

**Callback Hell: Callback** Hell is essentially **nested callbacks stacked below one another forming a pyramid structure**. Every callback depends/waits for the previous callback, thereby making a pyramid structure that affects the readability and maintainability of the code.

How do you error handle in react JS?

Error handling with Error Boundaries — For class components. Error boundaries are the most straightforward and effective way to handle errors that occur within your React components. You can **create an error boundary component by including the life cycle method componentDidCatch(error, info) if you use class component**

**What is encryption and decryption?**

Encryption is the process by which a readable message is converted to an unreadable form to prevent unauthorized parties from reading it. Decryption is the process of converting an encrypted message back to its original (readable) format.

**Can we use promise instead of callback?**

Both callbacks and promises help make our code asynchronous. Making callbacks async can cause issues such as callback hell, so **to avoid this we can use promises instead**, doing this helps us avoid this pitfall while keeping our code async and neat.

**What is Promise in JavaScript in simple words?**

JavaScript is single threaded, meaning that two bits of script cannot run at the same time; they have to run one after another. A Promise is **an object that represents the eventual completion (or failure) of an asynchronous operation, and its resulting value**.

**JavaScript Promise Methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| resolve(value) | Returns a new Promise object that is resolved with the given value |
| catch() | Appends the rejection handler callback |
| then() | Appends the resolved handler callback |
| finally() | Appends a handler to the promise |

**What is difference between call bind and apply in JavaScript?**

Summary. **call : binds the this value, invokes the function, and allows you to pass a list of arguments.**

**apply : binds the this value, invokes the function, and allows you to pass arguments as an array**.

bind : binds the this value, returns a new function, and allows you to pass in a list of arguments.

**Why do we need call apply and bind?**

**apply() when you want to invoke the function immediately, and modify the context**. Call/apply call the function immediately, whereas bind returns a function that, when later executed, will have the correct context set for calling the original function. This way you can maintain context in async callbacks and events.