# **Inception(Notes):**

React is a library.

You can inject the React library through the CDN links.

React has two links ‘React’ and ‘React-dom’

React- is a Core library that can be used on react-native and some other library

React-dom - is for operating over the DOM elements.

**Writing first code on React:**

Step 1: Inject React CDN links into the code. React & React-dom

Step 2: Create element using ‘React.createElement(‘h1’, {}, ‘Hello World)

Step 3: Create root(i.e., where to inject the react code or where render the elements) using the ReactDOM.createRoot(document.getElementById(‘rootElement’)

Step 4: Place the created element inside the root by ‘root.Render(heading)

A react element is a normal object. ( “ React.createElement(‘h1’, {}, ‘Hello World) ” will create an object using the arguments)

Homework: Browse the internet for CDN, Emmet, and cross-origin, what is async defer?

React will override all the elements inside the root.

# **Home-Tasks:**

## **What is emmet?**

Emmet is a web development toolkit and a shorthand language for writing HTML, XML, and CSS code more efficiently. It allows web developers to write code faster and with fewer keystrokes. Emmet is often used in code editors and integrated development environments (IDEs) to streamline the creation of web documents.

Some of the key features of Emmet include:

* Abbreviations: You can use short abbreviations to expand into complete HTML or CSS code. For example, you can type div>ul>li and then expand it into nested HTML elements.
* Nesting: You can quickly create nested elements by using > to indicate parent-child relationships.
* Multiplication: You can duplicate elements using an asterisk (\*) and specify how many times you want to duplicate an element.
* Attributes: You can set attributes for elements within the abbreviation. For instance, a[href="#"] generates an anchor tag with an href attribute.
* ID and Class Names: You can specify IDs and class names using the # and . characters, respectively.
* Customization: Emmet often allows you to create custom snippets, abbreviations, and templates for frequently used code patterns.

Emmet is supported by many popular code editors and IDEs, including Visual Studio Code, Sublime Text, and Atom. It can significantly speed up the coding process and improve code consistency for web developers.

## **What is the difference between a library and a framework?**

Libraries and frameworks are both tools used in software development, but they serve different purposes and have distinct characteristics. Here's the fundamental difference between the two:

* Library:
  + A library is a collection of functions, classes, or modules that you can use in your code to perform specific tasks or operations. Libraries provide a set of well-defined functions or methods that you can call as needed in your application.
  + You have control over the flow of your program, and you choose when and how to use the library's functionality.
  + Libraries are often designed to be reusable components that help you accomplish common tasks more easily. They are typically more focused and provide specific functionality.
  + Libraries do not dictate the overall structure of your application; you can use them as building blocks within your codebase.
* Example: jQuery is a library for working with HTML documents and providing various utilities for DOM manipulation and AJAX requests.
* Framework:
  + A framework, on the other hand, is a more comprehensive and structured software environment. It provides a set of rules, conventions, and a predefined architecture for building applications.
  + In a framework, you write code that fits into the framework's structure and follows its design patterns. The framework controls the flow of your application.
  + Frameworks often encompass a wide range of features and components, including libraries, but they also provide a blueprint for your application's overall structure.
  + Frameworks can be opinionated, meaning they enforce a specific way of doing things to ensure consistency and maintainability.
* Example: Ruby on Rails is a web application framework that defines the architecture of a web application, including routing, models, views, and controllers.

In summary, libraries are more like tools that you can use on an as-needed basis to perform specific tasks, while frameworks are more like the skeletons or foundations of your application, providing a structured environment and dictating how you should organize and build your application. The choice between using a library or a framework depends on the requirements of your project and your development style.

## **3. What is CDN? Why do we use it?**

CDN stands for "Content Delivery Network." It is a network of distributed servers (referred to as "edge servers" or "CDN nodes") that work together to deliver web content, such as text, images, videos, scripts, and other resources, to users based on their geographic locations. CDNs are designed to enhance the performance, reliability, and availability of web content, and they are widely used to optimize the delivery of web applications.

Here are some key aspects of CDN:

* Content Distribution: CDNs replicate and store copies of web content on multiple servers strategically placed in various geographical locations. When a user requests content, the CDN serves the content from the nearest edge server, reducing latency and speeding up content delivery.
* Load Balancing: CDNs automatically distribute incoming web traffic across multiple servers to balance the load. This helps prevent any single server from becoming overwhelmed with traffic, ensuring better performance and reliability.
* Caching: CDNs use caching to store static assets like images, scripts, and stylesheets closer to the end-users. Cached content can be quickly retrieved, reducing the load on the origin server and improving page load times.
* Security: CDNs often include security features like DDoS protection, web application firewalls, and SSL encryption to protect websites and applications from cyberattacks and threats.
* Scalability: CDNs are highly scalable and can handle traffic spikes and increased demand, making them suitable for high-traffic websites and applications.
* Content Optimization: CDNs can optimize content delivery by compressing files, resizing images on the fly, and delivering different versions of content based on the user's device and browser capabilities.
* Global Reach: CDNs have a global presence, which means that content can be delivered quickly to users in various regions around the world.

CDNs are used by a wide range of organizations, including e-commerce websites, media streaming platforms, content providers, and web applications, to ensure a fast and reliable experience for their users. They help reduce server load, minimize latency, and enhance the overall performance of web services. Popular CDN providers include Akamai, Cloudflare, Amazon CloudFront, and many others.

## **4. Why is React known as React?**

React is known as "React" because the name reflects its core philosophy and primary functionality. The name "React" is derived from its approach to building user interfaces: it reacts to changes in data and automatically updates the user interface in response to those changes. Here's why the name is fitting:

* Reactivity: React is designed to be a reactive library. It efficiently updates and re-renders the user interface components when the underlying data changes. This reactivity is achieved through a virtual DOM (Document Object Model) and a reconciliation algorithm, which minimizes the need to manually manipulate the DOM, allowing developers to describe how the UI should look based on the data.
* Component-Based: React encourages a component-based architecture, where the user interface is broken down into reusable components. Each component can be thought of as an independent unit that reacts to changes in its props and state, rendering itself accordingly. React's "reactive" nature is well-suited for this component-based approach.
* One-Way Data Flow: React primarily follows a one-way data flow, which means data flows from parent components to child components. When data changes at the top of the component tree, React efficiently propagates the changes downward, updating only the affected components. This approach is inherently reactive and contributes to React's name.
* Declarative Approach: React encourages a declarative style of defining how the UI should look based on the current data state. You declare what you want the UI to be, and React takes care of updating the actual UI to match that declaration, reacting to changes automatically.

In summary, the name "React" captures the essence of how the library operates: it reacts to changes in data and facilitates the creation of dynamic and interactive user interfaces. The name aligns with React's core principles of reactivity, component-based architecture, one-way data flow, and a declarative approach to UI development.

## **5. What is crossorigin?**

Certainly, let's explain the concept of "cross-origin" in simple terms with an example involving a user.

User Scenario: Imagine you are a user (let's call you "User A") who is browsing a website. This website contains some JavaScript code that's trying to load data from a different website (let's call it "Website B"). This is common when websites use external scripts or resources.

Now, "cross-origin" comes into play:

Cross-Origin Example:

* Without Cross-Origin Restrictions:
  + Imagine there are no cross-origin restrictions. In this case, "User A" can easily access and interact with data from any website, including Website B, regardless of where the data comes from.
  + For example, if Website A includes a script from Website B, that script can access data from Website B without any problems. No restrictions or barriers exist. The script can read and modify data on Website B as if it were part of the same website.

With Cross-Origin Restrictions:

* + In reality, cross-origin restrictions exist in web browsers to enhance security. These restrictions prevent scripts on one website (Website A) from freely accessing or modifying data on a different website (Website B).
  + So, if Website A tries to include a script from Website B, the browser will apply cross-origin security rules. This means that, by default, the script from Website A cannot read or modify data on Website B.
  + To make cross-origin requests work, Website B must explicitly grant permission by configuring its server to include the appropriate "Cross-Origin Resource Sharing" (CORS) headers. This is like giving User A permission to interact with Website B's data.

In Simple Terms:

Imagine you're User A on a website, and there's a rule that prevents that website's code from reading or changing data on another website. It's like a security guard that says, "You can't go in there and play with their stuff unless they say it's okay."

To allow User A (or the website) to play with the other website's stuff, the other website (Website B) has to say, "Sure, you can play with our stuff," by configuring its server with the right permissions.

So, "cross-origin" essentially means there are rules in place to keep different websites' code from freely messing with each other's stuff for security reasons, and these rules can be adjusted when needed for data sharing.

What is the difference between the React and React-DOM?

React- is a Core library that can be used on react-native and some other library. Eg.) React.createElement is working because of this(React core) library.

ReactDOM - is for operating over the DOM elements.Eg.) React.creatRoot is working because of this library(ReactDOM)

## **6. What is the difference between react.development.js and react.production.js files via CDN?**

When using React via a CDN (Content Delivery Network), you typically have the option to include either the development version (react.development.js) or the production version (react.production.js) of React in your web application. These two versions of React are optimized for different scenarios:

* react.development.js:
  + This is the development version of React.
  + It includes additional warning messages and development-specific features, which are helpful during the development and debugging phase.
  + It has a larger file size due to the additional code for warnings and development tools.
  + You should use this version while developing your application to get helpful error messages and development tools.
* react.production.js:
  + This is the production version of React.
  + It is optimized for performance and has all the development-specific code, warnings, and tools stripped out.
  + It is smaller in file size, making it more suitable for production use.
  + You should use this version in your production environment to reduce the size of your application and enhance performance.

The choice between the development and production versions of React depends on your deployment environment:

* Development Environment: Use react.development.js during development. It provides more informative error messages and warnings, which can help you identify and fix issues in your code.
* Production Environment: Use react.production.js in your production environment for the best performance and a smaller bundle size. This version doesn't include the extra code for debugging and warnings, which reduces the file size and improves application performance.

When deploying your application to production, it's common to have a build process that switches between the development and production versions based on your environment. For example, you might use the development version during development and testing and then switch to the production version when building your final deployment bundle. This ensures that you get the benefits of debugging tools during development while keeping your production code optimized for performance.

## **7. What are async and defer?**

# **Coding Tasks:**

Set up all the tools on your laptop

* vs Code
* Chrome
* Extensions of Chrome

Create a new Git repo

Build your first Hello World program using,

* Using just HTML
* Using JS to manipulate the DOM
* Using React

use CDN Links

Create an Element

Create nested React Elements

Use root.render

Push code to Github (Theory as well as code)

Learn about Arrow Functions before the next class