Emmet

Its way of writing HTML and CSS faster
It can be defined as bunch of different shortcuts and snippets where we type small amount of information and after we press enter it will generate a bunch of code

Ex-

1) #header(press Enter)

Then the code will be automatically generated

2) .title(press Enter)

Then the code will be automatically generated

Check this documentation for further shortcuts https://docs.emmet.io/cheat-sheet/

Async and Defer

By using async attribute in script tag HTML parsing goes on and script tags are fetched from the network in parallel when scripts are fetched and available in the browser after that scripts are executed only then HTML parsing resumes.
Till the time the scripts are executed in the browser HTML parsing is blocked.

By using defer attribute in script tag HTML parsing parsing goes on and script tags are fetched from the network in parallel ,script tags are only executed only when HTML parsing is fully complete.

Crossorigin

There may be requirement we need to fetch some data(text/images/videos) from different server to fetch these resources

Ex- We are in one Domain and we are making requests from somewhere else. The browser will decide how to handle that situation ,if we request some data from another server that server might not have data access to other servers.
Crossorigin attribute which we put inside of all these elements is a way to control or at least inform the browser as to how it should approach using these resources

There are two values we can pass in inside crossorigin attribute:

1) Anonymous

Ex- <script src=”/server_url” crossorigin=”anonymous”></script>

If we pass anonymous from the browser none of the cookies, none of the session data or any kind of information is sent to the server to receive the data from the server it will send data based on filtration done in the server as when none of the information is sent as a response.

2) use-credentials

Ex- <script src=”/server_url” crossorigin=”anonymous”></script>

If we pass user-credentials from the browser some information will be sent to the server based on the information sent to the server the server will filter the data and send it as a response.

Arrow Functions

When using arrow function instead of normal function there is small syntax difference

Normal Function Syntax

function getNumber(){

}

Arrow Function Syntax

const getNumber = () =>{

}

The main difference comes in using **this** inside Arrow function

```
let user = {
 name: "sandeep",
 age: 27,
```

```
 childObj: {
   lastName: "Mukherjee",
   getDetails() {
     console.log(this.lastName, "and", this.name);
   },
 },
};

user.childObj.getDetails();
```

o/p - Mukherjee and undefined

Reason - here this is pointing to childObj it has lastName so it printed lastName and name is not present inside childObj so it returned undefined

```
let user = {
 name: "sandeep",
 age: 27,

 getDetails() {
   const nestedArrowFunction = () => console.log(this.name);
   nestedArrowFunction();
 },
};

user.getDetails();
```

o/p- sandeep

Reason - here its takes the value of this from its parent function so here this points to user

Difference between Production and Development Env

Production env Build has ugly, minified(compressed) version of your javascript code, so this makes rendering of files on the end user's browser very quick and performance enhancing.

Development env build code is where we develop and add new features to the application and test it locally not on server speed will be little less than Production env due to the above reasons.

Why do we put script tag at the end of the body