**HTML**

**Q1. What is HTML?**

HTML stands for Hyper Text Markup Language. It is a language of the World Wide Web. It is a standard text formatting language which is used to create and display pages on the Web. HTML makes the text more interactive and dynamic.

### ****Q2. What are Attributes and how do we use them?****

### Each tag has additional attributes that change the way the tag behaves or is displayed. For example, a ****<input>**** tag has a type of attribute, which we can use to specify whether it’s a text field, checkbox, radio button or one of many more options.

### ****Q3. How to create a Hyperlink in HTML?****

The HTML provides an anchor tag to create a hyperlink that links one page to another page. These tags can appear in any of the following ways:

* **Unvisited link** – It is displayed, underlined and blue.
* **Visited link** – It is displayed, underlined and purple.
* **Active link** – It is displayed, underlined and red.

The **syntax** of Hyperlink in HTML is:

|  |  |
| --- | --- |
| 1 | <**a** href = "..........."> Link Text </**a**> |

### ****Q4. Name some common lists that are used when designing a page.****

There are many common lists used for design a page. We can choose any or a combination of the following list types:

* **Ordered list** – The ordered list displays elements in a numbered format. It is represented by <ol> tag.
* **Unordered list** – The unordered list displays elements in a bulleted format. It is represented by <ul> tag.
* **Definition list** – The definition list displays elements in definition form like in a dictionary. The <dl>, <dt> and <dd> tags are used to define description list.

### ****Q5. What is semantic HTML?****

### **Semantic html is the tag which define the meaning of the content they contain. For example:- <article/>, <header/> and <footer/> these are semantic html tag.**

### ****Q6. How to create a nested webpage in HTML?****

The HTML **iframe** tag is used to display a nested webpage. In other words, it represents a webpage within a webpage. The HTML <iframe> tag defines an inline frame. For example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <!**DOCTYPE** html>  <**html**>  <**body**>  <**h2**>HTML example</**h2**>  Use the height and width attributes to specify the size of the iframe:  <**iframe** src="<a href="https://www.edureka.co/">https://www.edureka.co/</a>" height="300" width="400"></**iframe**>  </**body**>  </**html**> |

### ****Q7. What is an image map?****

An image map is used for linking many different web pages using a single image. It is represented by <map> tag. We can define shapes in images that we want to include as part of an image mapping.

### ****Q8. Does a hyperlink only apply to text?****

No, hyperlinks can be used both on **texts** and **images**. The HTML anchor tag defines a hyperlink that links one page to another page.

### ****Q9. What is a marquee?****

Marquee is used for the scrolling text on a web page. It scrolls the image or text up, down, left or right automatically. We should put the text which we want to scroll within the**<marquee>……</marquee>** tag.

## **Q10. What are the tags used to separate a section of texts?**

There are three tags that can be used to separate the texts:

* **<br>** tag – Usually <br> tag is used to **separate** the **line of text**. It breaks the current line and conveys the flow to the next line
* **<p>** tag – This contains the **text** in the form of a new **paragraph**.
* **<blockquote>** tag – It is used to define a large quoted section. If we have a large quotation, then put the entire text within **<blockquote>……….</blockquote>** tag.

### ****Q11. What is the difference between DIV and SPAN in HTML?****

The difference between **span** and **div** is that a span element is **in-line** and usually used for a small chunk of HTML inside a line,such as inside a paragraph. Whereas a div or division element is **block-line** which is equivalent to having a line-break before and after it and used to group larger chunks of code.

### ****Q12. What is the purpose of using alternative texts in images?****

The purpose of using alternative texts is to define what the image is about.

### ****Q13. Is the <!DOCTYPE html> tag considered as a HTML tag?****

No, the <!DOCTYPE html> declaration is not an HTML tag.

There are many types of HTML, such as, HTML 4.01 Strict, HTML 4.01 Transitional, HTML 4.01 Frameset, XHTML 1.0 Strict, XHTML 1.0 Transitional, XHTML 1.0 Frameset, XHTML 1.1 etc. So, <!DOCTYPE html> is used to instruct the web browser about the HTML page.

### ****Q14. What are the entities in HTML?****

The HTML character entities are used as a replacement for reserved characters in HTML. We can also replace characters that are not present on wer keyboard by entities. These characters are replaced because some characters are reserved in HTML.

### ****Q15. Can we create a multi-colored text on a web page?****

Yes, we can create a multi-colored text on a web page.  To create a multicolor text, we can use <font color =”color”> </font> for the specific texts that we want to color.

**Q16. What is the use of a span tag? Explain with example.**

The span tag is used for following things:

* For adding color on text
* To add background on text
* Highlight any color text.

**Q17. Explain The Key Differences Between LocalStorage And SessionStorage Objects.**

The key differences between localStorage and sessionStorage objects are as follows:

* The localStorage object stores the data without an expiry date. However, **sessionStorage** object stores the data for only one session.
* In the case of a localStorage object, data will not delete when the browser window closes. However, the data gets deleted if the browser window closes, in the case of sessionStorage objects.
* The data in sessionStorage is accessible only in the current window of the browser. But, the data in the localStorage can be shared between multiple windows of the browser.

### ****Q18. What happens if we open the external CSS file in a browser?****

When we try to open the external CSS file in a browser, the browser cannot open the file, because the file has a different extension. The only way to use an external CSS file is to reference it using **<link/>** tag within another HTML document.

### ****Q19. What is the advantage of grouping several checkboxes together?****

The checkboxes don’t affect one another. But grouping these checkboxes together help to organize them. **Checkbox** buttons can have their name and do not need to belong to a group. A single web page can have many different groups of checkboxes.

### ****Q20. What is SVG?****

**HTML SVG** is used to describe the two-dimensional vector and vector or raster graphics. SVG images and their behaviors are defined in XML text files. So as XML files, we can create and edit an **SVG image** with the text editor. It is mostly used for vector type diagrams like pie charts, 2-Dimensional graphs in an X, Y coordinate system.

**Q21. List the media types and formats supported by HTML.**

HTML supports a wide range of media formats for sound, music, videos, movies, and animations. Some of the extensions supported by each media format are:

* **Images**– png, jpg, jpeg, gif, apng, svg, bmp, BMP ico, png ico
* **Audio**– MIDI, RealAudio, WMA, AAC, WAV, Ogg, MP3, MP4
* **Video**– MPEG, AVI, WMV, QuickTime, RealVideo, Flash, Ogg, WebM, MPEG-4 or MP4.

### ****Q22. What is Cell Spacing and Cell Padding?****

Cell Spacing is referred to as the space or gap between the two cells of the same table. Whereas Cell Padding is referred to as the gap or space between the content of the cell and cell wall or cell border.

**CSS**

### ****Q1. What is the full form of CSS?****

CSS stands for Cascading Style Sheets. It is a technology developed by the World Wide Web Consortium or W3C. It was developed to streamline the styling of webpages into a separate technology.

**Q2. What are the different ways we could integrate CSS into wer HTML page?**

There are three ways that we could integrate a certain CSS style:

1. We can integrate wer style using the style-tags in the head section of wer HTML page.
2. We can integrate wer style using inline styling.
3. We can write wer CSS in a separate file and add it to wer HTML page using the link tag.

**Q3. What are the advantages of using CSS?**

Following are the advantages of using CSS:

* The style of several documents can be controlled from a single site by using them.
* Multiple HTML elements can have many documents, where classes can be created.
* To group styles in complex situations, selector and grouping methods are used.

**Q4. What are the disadvantages of using CSS?**

Following are the disadvantages of using CSS:

* Ascending by selectors is not possible.
* Limitations of vertical control
* No expressions
* No column declaration
* Pseudo-class not controlled by dynamic behavior.
* Rules, styles, targeting specific text not possible.

### ****Q5. What is the difference between the usage of an ID and a Class?****

**ID** – An ID is unique. A particular ID can be only assigned to a single element. IDs are used when specific styling is being tried to be achieved over a single element.

**Class** – Just like the word suggests, a class is a collective way of targetting HTML elements for styling. Classes are not unique and multiple elements can have the same class. In fact, multiple classes can also be added to the same element to achieve the desired style and look.

**Q6. What are the ways to assign a certain colour to an element in CSS?**

CSS can assign a wide range of colours to elements using different notations. There are three notations as of now that are used that are explained below:

* **Hexadecimal notation**  
  A color in hexadecimal string notation always begins with the character **“#”**. After that, the hexadecimal digits of the color code is written. The string is case-insensitive.
* **RGB functional notation**RGB (Red/Green/Blue) functional notation, like hexadecimal string notation, represents colours using their red, green, and blue components (as well as, optionally, an alpha channel component for opacity). However, instead of using a string, the colour is defined using the CSS function**RGB()**. This function accepts as its input parameters the values of the red, green, and blue components and an optional fourth parameter, the value for the alpha channel.
* **HSL functional notation**Designers and artists often prefer to work using the HSL (Hue/Saturation/Luminosity) color method. On the web, HSL colors are represented using HSL functional notation. The HSL() CSS function is very similar to the RGB() function in usage otherwise.

**Q7. Explain the CSS Box Model and its different elements.**

The CSS box model is essentially a box that wraps around every HTML element. It consists of: content, padding, borders and margins.

* **Content** - The content of the box, where text and images appear.
* **Padding** - Clears an area around the content. The padding is transparent,
* **Border** - A border that goes around the padding and content.
* **Margin** - Clears an area outside the border. The margin is transparent.

### ****Q8. What is the z-index in CSS?****

The z-index helps specify the stack order of positioned elements that may overlap one another. The z-index default value is zero and can take on either a positive or negative number.

An element with a higher z-index is always stacked above than a lower index.

### ****Q9. What are CSS Sprites?****

CSS sprites combine multiple images into one single larger image. It is a commonly used technique for icons (Gmail uses it).

### ****Q10. What are pseudo-class in CSS?****

### **A pseudo- class is used to define a special state of the element. It can use to**

### **For example: - 1) style an element when a user mouses over it.**

### **2) style visited and unvisited link.**

### **3) style an element when it gets focus.**

### Q 10.a) what are pseudo-elements?

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

### For example: - is can be used to:

### Style the first letter or line of an element.

### Insert content before or after the content of an element.

### ****Q11. What is the float property used for in CSS?****

The float CSS property places an element on the left or right side of its container, allowing text and inline elements to wrap around it. The element is removed from the normal flow of the page, though it still remains a part of the flow (in contrast to absolute positioning). Below is the usage of float

float: none;

float: left;

float: right;

**Q12. What are the different media types allowed by CSS?**

There are four types of @media properties (including *screen*):

* **all** – for all media type devices
* **print** – for printers.
* **speech** – for screen readers that “reads” the page out loud
* **screen** – for computer screens, tablets, smart-phones etc.

### ****Q13. How do we control image repetition using CSS?****

We can use the **background-repeat** property to control image.

### ****Q14. What is the overflow property in CSS used for?****

The overflow property specifies whether to clip the content or to add scrollbars when an element’s content is too big to fit in a specified area. Below are the overflow options available in CSS –

overflow: visible; Default. The overflow is not clipped. The content renders outside the element’s box

overflow: hidden; The overflow is clipped, and the rest of the content will be visible.

overflow: scroll; The overflow is clipped, and a scroll bar is added to see the rest of the content.

overflow: auto; it is similar to the scroll, but it adds scroll bar only when necessary.

### ****Q15. What is responsive web design?****

**Responsive design** is an approach to web page creation that makes use of flexible lawets, flexible images and cascading style sheet media queries. The goal of **responsive design** is to build web pages that detect the visitor’s screen size and orientation and change the lawet accordingly.

### ****Q16. What is the difference between {visibility: hidden} and {display: none}?****

display:none removes the element from the documents. Its does not take any space on the page.

visibility:hidden hides the element but still take space in the layout, it just isn’t seen on the page.

### ****Q17. Explain the concept of specificity in CSS.****

**Specificity** is the means by which browsers decide which CSS property values are the most relevant to an element and, therefore, will be applied. Specificity is based on the matching rules which are composed of different sorts of CSS selectors.

### ****Q18. What is the use of box-shadow in CSS?****

The box-shadow CSS property adds shadow effects around an element’s frame. We can set multiple effects separated by commas. A box-shadow is described by X and Y offsets relative to the element, color, blur and spread radii. Below are a few implementations of box-shadow

box-shadow: 10px 5px 5px red;

box-shadow: 60px -16px teal;

box-shadow: 12px 12px 2px 1px rgba(0, 0, 255, .2);

box-shadow: inset 5em 1em gold;

### ****Q19. What is CSS flexbox?****

The flexbox layout officially called CSS**flexible box layout module** is a new layout module in CSS3. It makes easy to design flexible responsive layout structure without using float or positioning.

The prime characteristic of the flex container is the ability to modify the width or height of its children to fill the available space in the best possible way on different screen sizes.

Properties of flex box:

* flex-direction.
* flex-wrap.
* flex-flow.
* justify-content.
* align-items.

### ****Q20. What are the different ways to position a certain element in CSS?****

The position property specifies the type of positioning method used for an element.

There are five different position values:

position: fixed;

position: static;

position: absolute;

position: sticky;

position: relative;

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value.

**Q21. What’s the difference between a relative, fixed, absolute and statically positioned element?**

A positioned element is an element whose computed position property is either relative, absolute, fixed or sticky.

* **Static**  
  The default position; the element will flow into the page as it normally would. The top, right, bottom, left and z-index properties do not apply.
* **Relative**  
  The element’s position is adjusted relative to itself, without changing the lawet (and thus leaving a gap for the element where it would have been had it not been positioned).
* **Absolute**  
  The element is removed from the flow of the page and positioned at a specified position relative to its closest positioned ancestor if any, or otherwise relative to the initial containing block. Absolutely positioned boxes can have margins, and they do not collapse with any other margins. These elements do not affect the position of other elements.
* **Fixed**  
  The element is removed from the flow of the page and positioned at a specified position relative to the viewport and doesn’t move when scrolled.
* **Sticky**  
  Sticky positioning is a hybrid of relative and fixed positioning. The element is treated as relative positioned until it crosses a specified threshold, at which point it is treated as fixed positioned.

**JavaScript**

### 1. What exactly do we know about JavaScript?

It is basically an object-oriented programming language that is having a lot of applications. It can be used for both server-side, as well as client-side scripting and the good thing is it is known to provide the best results in every aspect. When it comes to building HTML WebPages, this language is often considered and is known to bring the most desired outcomes.

### 2. On a client machine, how we will detect the server using JavaScript?

For this, task, the users can simply proceed with the navigator.app string. The string performs this task automatically and the user just needs to define it at the right time. Also, the users need to make sure that the instructions are executed in the right manner. Sometimes there is a need to define the IP or other address of the server that needs to be detected.

### 3. If the argument is not a number, which function we will use in JavaScript?

isNan function is useful in case the argument is not a number. Actually, it remains true if the argument doesn’t have a number while on the other side it remains false if it is.

**4. How many data types are there in JavaScript? Can we name them?**

There are 6 data types in JavaScript, and they are:

* Function
* String
* Undefined
* Number
* Boolean
* Object

These data types are very useful, and they have applications in almost every important task performed by JavaScript.

### 7. How can we break the JavaScript code into different lines?

This can be done with the help of a backlash. However, we need to use it at the end of the initial line otherwise it will show an error at the end. The good thing is there is no limit on lines that can be created for the purpose of accommodating data. Also, doing this is not at all a big deal and one can easily handle this task.

### 8. Tell something about the undeclared and undefined variables?

Basically, there are a lot of programs that actually don’t exist in a program and are thus known as undeclared ones. There is always a runtime error displayed on the screen in case the program proceeds with reading the value of a variable that is not declared.

On the other side, declared variables as the name indicates are those which are well defined within a program but haven’t been assigned any value by the operator. The value is always returned in case of the program proceeds with reading them.

### 9. Are we familiar with the concept of negative infinity in JavaScript?

Yes, it is actually a number present in this language and can simply be derived by dividing any number having a negative value with zero. There are basically some important applications of negative infinity in JavaScript.

### 10. What are timers in JavaScript? What do we know about them?

There is a provision in JavaScript for the programmers to set a specific time when it comes to executing a code. This can be done with the help of timers. It is actually a similar phenomenon as we set an alarm on wer cell phone. The code can also be repeated at a specific time once executed. There are functional such as setInterval, clear internal that can be considered for this.

### 11. What exactly is the difference between Session State and the View State in JavaScript?

Session State is an option that can be considered to access all the pages within a web application while on the other side View State is used for accessing a session only.

### 12. In JavaScript, it is possible for the programmers to convert the string to an integer?

Yes, this can be done easily in JavaScript. For this, the function parseInt () is used. It generally takes the string into the first parameter and then to the next parameter which is considered as the base of given string.

**13. Name the two basic groups of data types in JavaScript?**

These are:

* Reference types
* Primitive

### 14. When can we use NULL in JavaScript?

Basically, it is used when there is no object or there is no value is there on a string. It let the array proceed smoothly and perform its task very reliably. There is no need for the users to worry while executing this as it doesn’t show the runtime error at any stage. Also, this can be used in special cases to check whether all the strings have defined variables or numbers present with them or not.

### 15. How the button configuration in the Confirmation box is different from the Alert box in Javascript?

The basic difference is there are two buttons in the Conformation box, and they are named OK and Cancel. On the other side, the Alert box contains only one button and i.e. OK. In addition to this, there are other differences in the configuration, and they are hidden in a few cases for security purposes.

### 16. What exactly are the escape characters?

The Escape character is used in such a case. Users can make it display again anytime and this can be done with the help of a backlash. Backlash can simply be placed before the character and this makes them display in a proper manner.

### 17. In JavaScript, which method we will find similar to the pop method?

The Pop method is very much similar to the shift method but there are some noticeable differences. In the pop method, the last element of the defined array is taken generally, and then the same is returned. This often cut down the size of the array. On the other side, the Shift method generally considers the first element of the array or any other element in the beginning.

### ****Q18. What are the features of JavaScript?****

* It is a **lightweight, interpreted** programming language.
* It is designed for creating **network-centric** applications.
* It is complementary to and **integrated** with Java.
* It is an **open** and **cross-platform** scripting language.

**Q19. How can we create an object in JavaScript?**

JavaScript supports **Object** concept very well. We can create an object using the **object literal** as follows –

Var emp = {

Name: “Daniel”,

Age: 12

}

### ****Q20. How can we create an Array in JavaScript?****

We can define arrays using the **array literal** as follows-

Var x = []’

Var x = [1,2,3,6];

### ****Q21. What is a name function in JavaScript & how to define it?****

A named function declares a name as soon as it is defined. It can be defined using **function** keyword as :

Function] named(){

//write code here

}

### ****Q22. Define anonymous function.****

In JavaScript, an anonymous function is a function that does not have a name. It is typically used for short, one-off tasks where naming the function is not essential.

Function(parameter){

//body of the function

}

### ****Q23. What are the scopes of a variable in JavaScript?****

The scope of a variable is the **region** of wer program in which it is **defined**. JavaScript variable will have only two scopes.  
• **Global Variables** − A global variable has global scope which means it is visible everywhere in wer JavaScript code.  
•**Local Variables** − A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

### ****Q24. What is the purpose of ‘This’ operator in JavaScript?****

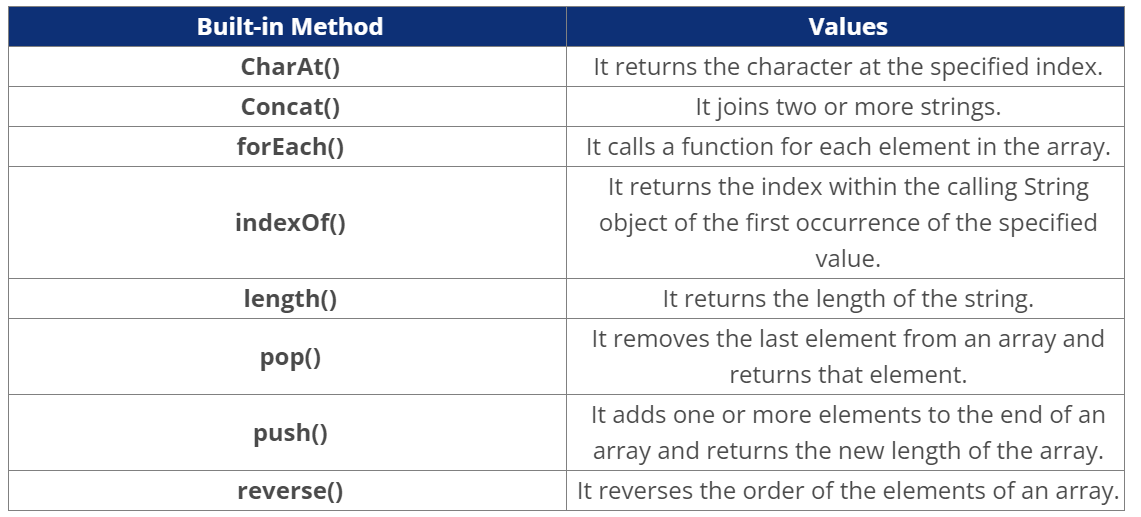
In JavaScript, this keyword refers to the object it belongs to. Depending on the context, this might have several meanings. This pertains to the global object in a function and the owner object in a method, respectively.

### ****Q25. What is Callback?****

A simple JavaScript function that is sent as an option or parameter to a method is called a callback. The function is called “call back” because it is meant to be invoked after another function has completed running. Functions are objects in JavaScript. This means that other functions can return functions and accept functions as arguments.

### ****Q26. What is Closure? Give an example.****

**Closures** are created whenever a variable that is defined outside the **current scope** is accessed from within some inner scope. It gives we access to an outer function’s scope from an inner function. In JavaScript, closures are created every time a function is created. To use a closure, simply define a function inside another function and expose it.

**Q27. Name some of the built-in methods and the values returned by them.**

**Q28. What are the variable naming conventions in JavaScript?**

The following **rules** are to be followed while **naming variables** in JavaScript:

1. We should not use any of the JavaScript **reserved keyword** as variable name. For example, break or boolean variable names are not valid.
2. JavaScript variable names should not start with a **numeral** (0-9). They must begin with a letter or the underscore character. For example, 123name is an invalid variable name but \_123name or name123 is a valid one.
3. JavaScript variable names are **case sensitive**. For example, Test and test are two different variables.

### ****Q29. How does Type Of Operator work?****

The **type of** operator is used to get the data type of its operand.

### ****Q30. How to create a cookie using JavaScript?****

The simplest way to create a cookie is to assign a string value to the **document.cookie** object, which looks like this-

Document.cookie = “key1 = value1; key2 = value2; expires = date”;

### ****Q31. How to delete a cookie using JavaScript?****

If we want to delete a cookie so that subsequent attempts to read the cookie in JavaScript return nothing, we just need to set the expiration date to a time in the past. We should define the cookie path to ensure that we delete the right cookie. Some browsers will not let we delete a cookie if we don’t specify the path.

**Q32. Explain call(), apply() and, bind() methods.**

1. `**call()`:** The call method is used to call a function and specify the “this” value for the function. It takes two arguments: the value to be used as the “this” value inside the function, and an optional list of arguments to be passed to the function.

function greet(name) {

  console.log (`Hello, ${name}! My name is ${this.name}. `);

}

const person = {

  name: ‘John’

};

greet.call(person, ‘Alice’);

// Output: Hello, Alice! My name is John.

“`

1. `**apply()`:** The `apply()` method is similar to `call()`, but it accepts arguments as an array or an array-like object. It also allows we to set the `this` value explicitly.

“`javascript

function greet (name, age) {

  console.log (`Hello, ${name}! I am ${age} years old. `);

  console.log (`My name is ${this.name},`);

}

const person = {

  name: ‘John’

};

greet.apply(person, [‘Alice’, 25]);

// Output: Hello, Alice! I am 25 years old.

//         My name is John.

“`

1. `**bind()`:** The bind method is used to create a new function with the same body as the original function, but with a different “this” value. It also takes an optional list of arguments to be passed to the new function when it is called. “

function greet() {

  console.log (`Hello, ${this.name}!`);

}

const person = {

  name: ‘John’

};

const greetPerson = greet.bind(person);

greetPerson();

// Output: Hello, John!

“`

**Q33. Explain Hoisting in javascript**.

Hoisting is a behavior in JavaScript where variable and function declarations are moved to the top of their respective scopes during the compilation phase, before the actual code execution takes place. This means that we can use variables and functions before they are declared in wer code.

However, it is important to note that only the declarations are hoisted, not the initializations or assignments. So, while the declarations are moved to the top, any assignments or initializations remain in their original place.

In the case of variable hoisting, when a variable is declared using the `var` keyword, its declaration is hoisted to the top of its scope. This means we can use the variable before it is explicitly declared in the code. However, if we try to access the value of the variable before it is assigned, it will return `undefined`.

“`javascript

console.log(myVariable);  // Output: undefined

var myVariable = 10;

console.log(myVariable);  // Output: 10

“`

In the above example, even though `myVariable` is accessed before its declaration, it doesn’t throw an error. However, the initial value is `undefined` until it is assigned the value of `10`.

Function declarations are also hoisted in JavaScript. This means we can call a function before it is defined in the code. For example:

“`javascript

myFunction();  // Output: “Hello, World!”

function myFunction() {

  console.log(“Hello, World!”);

}

“`

In this case, the function declaration is hoisted to the top, so we can call `myFunction()` before its actual declaration.

**Q34. What is the difference between exec () and test () methods in javascript?**

The `exec()` and `test()` methods are both used in JavaScript for working with regular expressions, but they serve different purposes.

1. `exec()` method: The `exec()` method is a regular expression method that searches a specified string for a match and returns an array containing information about the match or `null` if no match is found. It returns an array where the first element is the matched substring, and subsequent elements provide additional information such as captured groups, index, and input string.
2. `test()` method: The `test()` method is also a regular expression method that checks if a pattern matches a specified string and returns a boolean value (`true` or `false`) accordingly. It simply indicates whether a match exists or not.

**Q35. What is the difference between the var and let keywords in javascript?**

The var and let keywords are both used to declare variables in JavaScript, but they have some key differences.

Scope

The main difference between var and let is the scope of the variables they create. Variables declared with var have function scope, which means they are accessible throughout the function in which they are declared. Variables declared with let have block scope, which means they are only accessible within the block where they are declared.

Hoisting

Another difference between var and let is that var declarations are hoisted, while let declarations are not. Hoisting means that the declaration of a variable is moved to the top of the scope in which it is declared, even though it is not actually executed until later.

Redeclaration

Finally, var declarations can be redeclared, while let declarations cannot. This means that we can declare a variable with the same name twice in the same scope with var, but we will get an error if we try to do the same with let.

**Q36. Why do we use the word “debugger” in javascript?**

“Debugger” in JavaScript is a tool or feature that helps writers find and fix mistakes in their code. It lets them stop the code from running, look at the variables, and analyze phrases to find bugs or strange behavior and fix them. The word “debugger” comes from the idea of getting rid of “bugs” or mistakes, which is how early engineers fixed problems with hardware.

**Q37. Explain Implicit Type Coercion in javascript.**

In JavaScript, implicit type coercion is when values are automatically changed from one data type to another while the code is running. It happens when numbers of different types are used in actions or comparisons.

### ****Q38. What is the difference between Attributes and Property?****

**Attributes-**  provide more details on an element like id, type, value etc.

**Property-**  is the value assigned to the property like type=”text”, value=’Name’ etc.

### ****Q39. List out the different ways an HTML element can be accessed in a JavaScript code.****

Here are the list of ways an HTML element can be accessed in a Javascript code:  
(i) **getElementById(‘idname’):** Gets an element by its ID name  
(ii) **getElementsByClass(‘classname’):** Gets all the elements that have the given classname.  
(iii) **getElementsByTagName(‘tagname’):** Gets all the elements that have the given tag name.  
(iv) **querySelector():** This function takes css style selector and returns the first selected element.

**Q40. Explain Higher Order Functions in javascript.**

Higher-order functions in JavaScript are functions that can take other functions as inputs or return functions as their outputs. They make it possible to use strong functional programming methods that make code more flexible, reused, and expressive.

### ****Q41. What is the difference between Local storage & Session storage?****

**Local Storage** – The data is not sent back to the server for every HTTP request (HTML, images, JavaScript, CSS, etc) – reducing the amount of traffic between client and server. It will stay until it is manually cleared through settings or program.

**Session Storage** – It is similar to local storage; the only difference is while data stored in local storage has no expiration time, data stored in session storage gets cleared when the page session ends. Session Storage will leave when the browser is closed.

### ****Q42. What is the difference between the operators ‘==‘ & ‘===‘?****

The main difference between “==” and “===” operator is that formerly compares variable by making **type correction** e.g. if we compare a number with a string with numeric literal, == allows that, but === doesn’t allow that, because it not only checks the value but also type of two variable, if two variables are not of the same type “===” return false, while “==” return true.

**Q43. What is currying in JavaScript?**

Currying is a JavaScript functional programming approach that converts a function with many parameters into a succession of functions that each take one argument. It allows us to use only a portion of the inputs, allowing us to create functions that may be used several times and are specialized.

### ****Q44. What is the difference between innerHTML & innerText?****

**innerHTML** – It will process an HTML tag if found in a string.

**innerText** – It will not process an HTML tag if found in a string

### ****Q45. How do JavaScript primitive/object types pass in functions?****

One of the differences between the two is that Primitive Data Types are passed By Value and Objects are passed By Reference.

**By Value** means creating a COPY of the original.

**By Reference** means creating an ALIAS to the original.

**Q 46) What is currying?**

Ans: - currying is a functional programming technique that involves breaking down a function that takes multiple arguments into a series of functions that take one argument each. This creates a chain of functions, where each function returns another function until the final result is achieved.

Currying in JavaScript can be for the following reasons:

* Currying is helpful in Event handling.
* By using the currying function, we can avoid passing the same variable many times.
* Currying in JavaScript can be used to make a higher-order function.

**Q 47) What is closure?**

Ans: - closure is a JavaScript features which allow the inner function to access the outer function. In JavaScript every time closure is created with the creation of functions.

Closure has 3 scopes.

* Access to its own scope.
* Access to the variables of outer function.
* Access to the global variable.

Example: -

Function outer(){

Let a = 10;

Function inner(){

Return a

}

Return inner();

}

Console.log(outer());

**Q 48) explain Promises in JavaScript.**

**Ans: -**

* In JavaScript, a Promise is an object that will produce a single value sometime in the future. If the promise is successful, it will produce a resolved value, but if something goes wrong then it will produce a reason why the promise failed.
* Promises is used to perform the asynchronous operation.
* In promise we have .then() method which executed when promise resolved and we have .catch() method which catch the error if promise is rejected.

JavaScript promises can be in one of three possible states. These states indicate the progress of the promise. They are:

* pending: This is the default state of a defined promise.
* fulfilled:  This is the state of a successful promise.
* rejected: This is the state of a failed promise.

**The**Promise.all ()**method:**

promise.all() accepts an array of promises as an argument but returns a single promise as the output. The single promise it returns resolves with an array of values if all the promises in the input array are fulfilled. The array Promise.all() resolves with will contain the resolve values of individual promises in the input array.

const promise1 = Promise.resolve(`First Promise's Value`);

const promise2 = new Promise((resolve, reject) =>

setTimeout(reject, 2000, `First reason for rejection`)

);

const promise3 = new Promise((resolve, reject) =>

setTimeout(reject, 3000, `Second reason for rejection`)

);

Promise.all([promise1, promise2, promise3]);

// Output on the console

// \*Promise {<rejected>: "First reason for rejection"}\*

**The**Promise.race()**method:**

Promise.race() accepts an array of promises as an argument and returns a single promise as an output. The single promise it returns is the fastest promise to finish running—resolved or not. This means Promise.race() will return the promise with the shortest execution time in an array of promises.

const promise1 = new Promise((resolve) =>

setTimeout(resolve, 3000, `First Promise's Value`)

);

const promise2 = new Promise((resolve) =>

setTimeout(resolve, 2000, `Second Promise's Value`)

);

const promise3 = Promise.resolve(`Third Promise's Value`);

Promise.race([promise1, promise2, promise3]);

// Output on the console

// \*Promise {<fulfilled>: "Third Promise's Value"}\*

**The**Promise.any()**method:**

Promise.any() accepts an array of Promises as an argument but returns a single Promise as the output. The single promise it returns is the first resolved promise in the input array. This method waits for any promise in the array to be resolved and would immediately return it as the output.

**React JS**

**1. What is React?**

* React is a front-end JavaScript library developed by Facebook in 2011.
* It follows the component based approach which helps in building reusable UI components.
* It is used for developing complex and interactive web and mobile UI.
* Even though it was open-sourced only in 2015, it has one of the largest communities supporting it.

### ****2. What are the features of React?****

Major features of React are listed below:

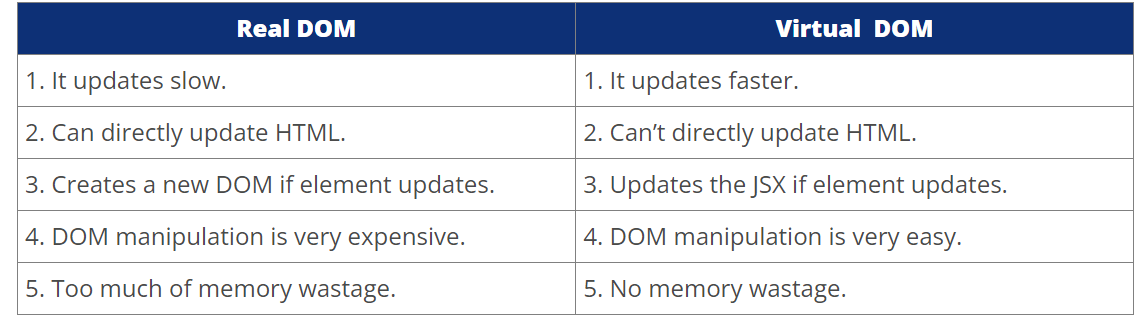
1. It uses the **virtual DOM** instead of the real DOM.
2. It uses **server-side rendering**.
3. It follows **uni-directional data flow** or data binding.

**3. What are the limitations of React?**

Limitations of React are listed below:

1. React is just a library, not a full-blown framework.
2. Its library is very large and takes time to understand.
3. It can be little difficult for the novice programmers to understand.
4. Coding gets complex as it uses inline templating and JSX.

**4.  Differentiate between Real DOM and Virtual DOM.**

****

**4. What do you understand by Virtual DOM? Explain its works.**

A virtual DOM is a lightweight JavaScript object which originally is just a copy of the real DOM. It is a node tree that lists the elements, their attributes and content as Objects and their properties.

This Virtual DOM works in three simple steps.

1. Whenever any underlying data changes, the entire UI is re-rendered in Virtual DOM representation.
2. Then the difference between the previous DOM representation and the new one is calculated.
3. Once the calculations are done, the real DOM will be updated with only the things that have actually changed.

### 6. ****Why can’t browsers read JSX?****

Browsers can only read JavaScript objects but JSX in not a regular JavaScript object. Thus to enable a browser to read JSX, first, we need to transform JSX file into a JavaScript object using JSX transformers like Babel and then pass it to the browser.

### ****7. How is React different from Angular?****

### 

### ****8. “In React, everything is a component.” Explain.****

Components are the building blocks of a React application’s UI. These components split up the entire UI into small independent and reusable pieces. Then it renders each of these components independent of each other without affecting the rest of the UI.

### ****9. What is the purpose of render() in React.****

Each React component must have a **render()**mandatorily. It returns a single React element which is the representation of the native DOM component. If more than one HTML element needs to be rendered, then they must be grouped together inside one enclosing tag such as **<form>, <group>,<div>** etc. This function must be kept pure i.e., it must return the same result each time it is invoked.

**10. How can you embed two or more components into one?**

We can embed components into one in the following way:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | **class** MyComponent **extends** React.Component{      render(){  **return**(    <div>    <h1>Hello</h1>                    <Header/>              </div>            );      }  }  **class** Header **extends** React.Component{      render(){  **return**    <h1>Header Component</h1>       };  }  ReactDOM.render(      <MyComponent/>, document.getElementById('content')  ); |

**11. What is Props?**

Props is the shorthand for Properties in React. They are read-only components which must be kept pure i.e. immutable. They are always passed down from the parent to the child components throughout the application. A child component can never send a prop back to the parent component. This help in maintaining the unidirectional data flow and are generally used to render the dynamically generated data.

**12. What is a state in React and how is it used?**

Basically, states are the objects which determine components rendering and behavior. They are mutable unlike the props and create dynamic and interactive components. They are accessed via this.state().

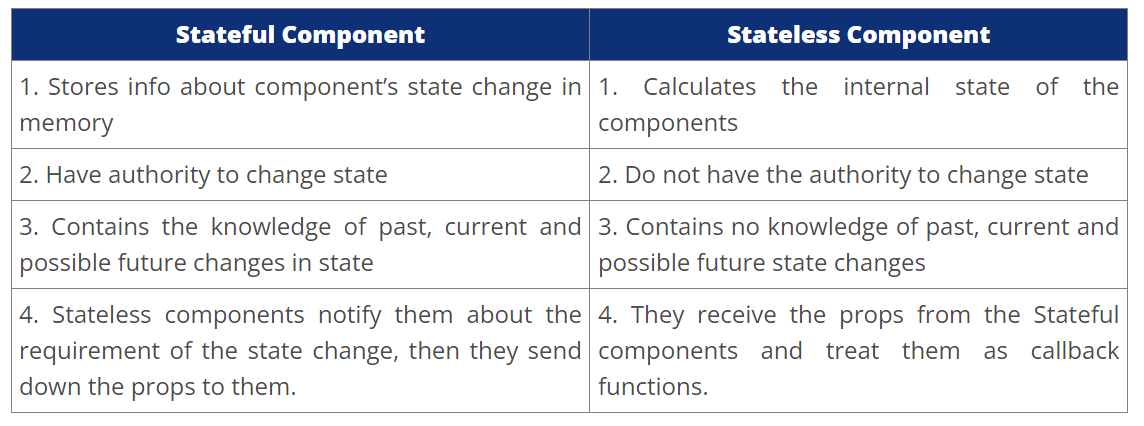
**13. What is arrow function in React? How is it used?**

Arrow functions are more of brief syntax for writing the function expression. They are also called ‘fat arrow‘ (=>) the functions. These functions allow to bind the context of the components properly since in ES6 auto binding is not available by default. Arrow functions are mostly useful while working with the higher order functions.

Syntax: - const functionName = () => {

//write code here

}

**Q14 Differentiate between stateful and stateless components.**

**15. What are the different phases of React component’s lifecycle?**

There are three different phases of React component’s lifecycle:

1. *Initial Rendering Phase:* This is the phase when the component is about to start its life journey and make its way to the DOM.
2. *Updating Phase:*Once the component gets added to the DOM, it can potentially update and re-render only when a prop or state change occurs. That happens only in this phase.
3. *Unmounting Phase:*This is the final phase of a component’s life cycle in which the component is destroyed and removed from the DOM.

**16. Explain the lifecycle methods of React components in detail.**

Some of the most important lifecycle methods are:

1. ***componentWillMount()***–Executed just before rendering takes place both on the client as well as server-side.
2. ***componentDidMount()***–Executed on the client side only after the first render.
3. ***componentWillReceiveProps()***– Invoked as soon as the props are received from the parent class and before another render is called.
4. ***shouldComponentUpdate()***–Returns true or false value based on certain conditions. If you want your component to update, return **true** else return **false**. By default, it returns false.
5. ***componentWillUpdate()***– Called just before rendering takes place in the DOM.
6. ***componentDidUpdate()***–Called immediately after rendering takes place.
7. ***componentWillUnmount()***– Called after the component is unmounted from the DOM. It is used to clear up the memory spaces.

**17. What is an event in React?**

In React, events are the triggered reactions to specific actions like mouse hover, mouse click, key press, etc. Handling these events are similar to handling events in DOM elements. But there are some syntactical differences like:

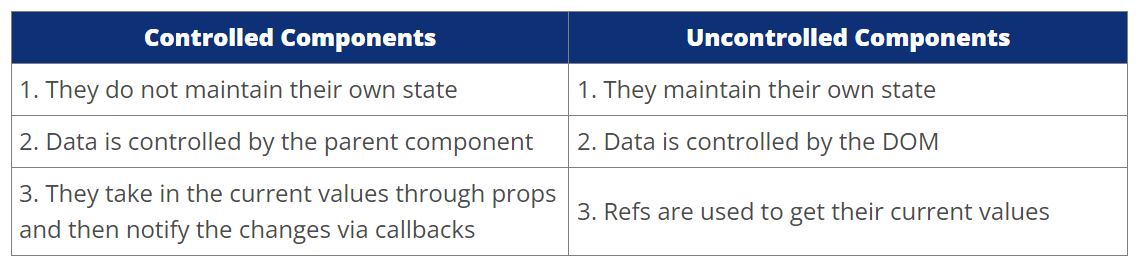
1. Events are named using camel case instead of just using the lowercase.
2. Events are passed as functions instead of strings.

**18. What are synthetic events in React?**

Synthetic events are the objects which act as a cross-browser wrapper around the browser’s native event. They combine the behavior of different browsers into one API. This is done to make sure that the events show consistent properties across different browsers.

**19. What do you understand by refs in React?**

Refs is the short hand for References in React. It is an attribute which helps to store a reference to a particular React element or component, which will be returned by the components render configuration function. It is used to return references to a particular element or component returned by render().

**20. What do you know about controlled and uncontrolled components?**

**21. What are Higher Order Components (HOC)?**

Higher Order Component is an advanced way of reusing the component logic. Basically, it’s a pattern that is derived from React’s compositional nature. HOC are custom components which wrap another component within it. They can accept any dynamically provided child component but they won’t modify or copy any behavior from their input components. You can say that HOC are ‘pure’ components.

**22. What are Pure Components?**

Pure components are the simplest and fastest components which can be written. They can replace any component which only has a render(). These components enhance the simplicity of the code and performance of the application.

**23. What were the major problems with MVC framework?**

Following are some of the major problems with MVC framework:

* DOM manipulation was very expensive.
* Applications were slow and inefficient.
* There was huge memory wastage.
* Because of circular dependencies, a complicated model was created around models and views.

**24. Explain Flux.**

Flux is an architectural pattern which enforces the uni-directional data flow. It controls derived data and enables communication between multiple components using a central Store which has authority for all data. Any update in data throughout the application must occur here only. Flux provides stability to the application and reduces run-time errors.

**25. What is Redux?**

Redux is one of the most trending libraries for front-end development in today’s marketplace. It is a predictable state container for JavaScript applications and is used for the entire applications state management. Applications developed with Redux are easy to test and can run in different environments showing consistent behavior.

**26. List down the components of Redux.**

Redux is composed of the following components:

1. **Action** – It’s an object that describes what happened.
2. **Reducer**– It is a place to determine how the state will change.
3. **Store** – State/ Object tree of the entire application is saved in the Store.
4. **View** – Simply displays the data provided by the Store.

**27. How are Actions defined in Redux?**

In Redux, actions are created using the functions called Action Creators. Below is an example of Action and Action Creator:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | function addTodo(text) {  **return** {                  type: ADD\_TODO,                   text      }  } |

**28. Explain the role of Reducer.**

Reducers are pure functions which specify how the application’s state changes in response to an ACTION. Reducers work by taking in the previous state and action, and then it returns a new state. It determines what sort of update needs to be done based on the type of the action, and then returns new values. It returns the previous state as it is, if no work needs to be done.

**29. What is the significance of Store in Redux?**

A store is a JavaScript object which can hold the application’s state and provide a few helper methods to access the state, dispatch actions and register listeners. The entire state/ object tree of an application is saved in a single store. As a result of this, Redux is very simple and predictable. We can pass middleware to the store to handle the processing of data as well as to keep a log of various actions that change the state of stores. All the actions return a new state via reducers.

**30. What are the advantages of Redux?**

Advantages of Redux are listed below:

* **Predictability of outcome –**Since there is always one source of truth, i.e. the store, there is no confusion about how to sync the current state with actions and other parts of the application.
* **Maintainability –**The code becomes easier to maintain with a predictable outcome and strict structure.
* **Server-side rendering –** You just need to pass the store created on the server, to the client side. This is very useful for initial render and provides a better user experience as it optimizes the application performance.
* **Developer tools –**From actions to state changes, developers can track everything going on in the application in real time.
* **Community and ecosystem –**Redux has a huge community behind it which makes it even more captivating to use. A large community of talented individuals contribute to the betterment of the library and develop various applications with it.
* **Ease of testing –**Redux’s code is mostly functions which are small, pure and isolated. This makes the code testable and independent.
* **Organization –**Redux is precise about how code should be organized, this makes the code more consistent and easier when a team works with it.

**31. What is React Router?**

React Router is a powerful routing library built on top of React, which helps in adding new screens and flows to the application. This keeps the URL in sync with data that’s being displayed on the web page.

**32. Why do we need a Router in React?**

A Router is used to define multiple routes and when a user types a specific URL, if this URL matches the path of any ‘route’ defined inside the router, then the user is redirected to that particular route. So basically, we need to add a Router library to our app that allows creating multiple routes with each leading to us a unique view.

**33. Explain the concept of a Hook in React.**  
“Hooks are a new feature in React that allows us to add state and other React features to functional components. They were introduced in React 16.8 and have since become a popular way to manage state and side effects in functional components. Hooks are named functions that start with the word use and allow us to reuse stateful logic across components without having to write a class component. For example, the useState Hook allows us to add state to a functional component and the useEffect Hook lets us perform side effects like data fetching or updating the document title. Hooks make our code more reusable, easier to understand, and easier to test.”