AN INTERVIEW GUIDE

UI INTERVIEW QUESTIONS & ANSWERS

ANGULAR,ANGULARJS, JAVASCRIPT,JQUERY, TYPESCRIPT,REST API, HTML5,CSS3,AJAX

BY SATYAPRIYA MISHRA

UI INTERVIEW QUESTIONS & ANSWERS

SATYAPRIYA MISHRA

Table of contents

About the book

This book is written solely for the UI/ Front end developers who are looking for an in-depth guide to hone their understanding of the front end technologies for cracking interviews to get their dream job. With the recent advent of numerous libraries and frameworks in the UI domain it has become increasingly difficult on developers' part to keep track of all the recent happenings in the tech world. To avoid information overloading from the readers this book covers only the most important interview questions and answers so that they can be grasped with very little time. It is always good to understand programming concepts through their real time applications and keeping in mind this fact most of the examples in this book are given keeping real time applications in mind. The concepts described in the book are explained with very simple terms without digging into technical jargons to suit both entry level and experienced developers. Developers who are not actively looking for a job can also benefit from the book by going through the concepts and examples. This book does not explicitly outlines theoretical concepts rather produces them in the form of questions and answers more specific to the standards of interviews. Hundreds of questions have been incorporated to make the guide the ultimate resource for developers for clearing any front end interview.

About the Author

Satyapriya Mishra is a full stack web developer with more than 5 years of industry experience whose technology kit bag includes but not limited to Angular, AngularJS, TypeScript, JavaScript, JQuery, ReactJS, VueJS, D3JS,NodeJS,.Net,PHP,Python,MySql, MongoDB, CosmosDB, LiteDB, Gulp, Webpack, HTML5, CSS3, Less, SCSS, Bootstrap etc. Currently he is guiding hundreds of job seekers to get jobs by means of training and counseling. When he is not coding he tries to spread a word or two about technology by regularly attending and speaking at meet-ups and workshops. He has to his credit a JavaScript library and a couple of chrome extensions and an open source book 'EcmaScript6 and Typescript with Examples' which was generously received by the readers. He can be reached at www.imsatya.com.

1. What is HTML?

Ans:-HTML stands for Hypertext Markup Language. It is s markup language for World Wide Web. HTML was developed by Tim Berners Lee during 1990's. It consists of tags which are used to define content and web page formatting.

2. What is DOCTYPE?

Ans:- DOCTYPE stands for Document Type Declarations. DOCTYPE is used to specify the web browsers that which types of documents(such as SGML or XML Documents) it will receive. <!DOCTYPE> is declared above the <html> tag. The browser detects the version of html from the Doctype declaration.

3. What are the different versions of HTML?

Ans:- HTML has many versions since it inceptions during 1990's. HTML version are HTML, HTML+, HTML 2.0, HTML 3.2, HTML 4.0 and the latest version HTML 5.

4. What are HTML Tags?

Ans:- HTML tags similar to keywords which has specifies markups, for example, for paragraph we can use HTML tag( ). HTML tags mostly comes in pair like, , .

5. What is HTML Attribute?

Ans:- HTML attribute adds additional information to the HTML Elements. For example, <font size="5" color="green">, here size and color are html attributes.Class and id are the most important attributes to select html nodes.

6. What is Hyperlinks?

Ans:- Hyperlinks are links to navigate from one page to another page or from one part of the page to some other part of the same web page. Mostly they are used for anchor tags.

7. What are the web standards?

Ans:- Web standards are standards specified for the internet or World Wide Web aspects for improving internet usability by all major OS and browsers.We can think of them as set of rules to keep in mind while developing web applications.

8. What is the HTML tag for line Break?

Ans:- HTML tag for break is
. The HTML break tag is used to insert a line break.

9. How to create Tables using HTML?

Ans:- Table can be created using the table tag like the below example.

```
<html>
    <head></head>
    <body>
        <table border = "2">
                <tr>
                    <th>Heading1</th>
                    <th>Heading1</th>
                    <th>Heading1</th>
                </tr>
                <tr>
                    <td>Some text</td>
                    <td>Some text</td>
                    <td>Some text</td>
                </tr>

        </table>
    </body>
</html>
```

And the output of the following code snippet as shown in the browser will be

Heading1	Heading1	Heading1
Some text	Some text	Some text

10. How to create an image tag ?

Ans:- The following code snippet will create an image tag and an image will be shown in the browser.

```
<img src = "myimage.jpg" title = "myimage">
```

11. How to open a link in new tab or window?

Ans:- To open a link in new tab or window, we have to use the following html code:

```
<a href=”http://www.mylink.com” target=”_blank>Click to visit to mylink.com!</a>
```

When we click on the link the browser will open the link in a new window.

12. What are the different types of Headings supported by HTML?

Ans:- HTML headings are important to highlight the contents of the documents. HTML supports 6 types of heading starting from <h1> to <h6>.

13. What are the types of HTML lists?

Ans:- There are two types of HTML list, they are ordered list and unordered list.

i) Ordered List:

The following code snippet demonstrates ordered list.

```
<ol type="a">
    <li>Item</li>
    <li>Item</li>
    <li>Item</li>
</ol>
```

And the output of the above code snippet will be like as follows.

a. Item
b. Item
c. Item

We can set the type of ordering by changing the type attribute in the ordered list as follows. Suppose we want a list with numbers as orders , then we can do the following.

```
<ol type="1">
    <li>Item</li>
    <li>Item</li>
    <li>Item</li>
</ol>
```

And the corresponding output will be :

1. Item
2. Item
3. Item

ii) Unordered List:

In unordered list the items are displayed as bullet ordered. The code for unordered list is as follows.

```
<ul>
    <li>Item</li>
    <li>Item</li>
    <li>Item</li>
</ul>
```

And the corresponding output will be:

- Item
- Item
- Item

We can remove the bullet marks from the unordered list by using **list-style-type: none;** property.

*******Unordered list is mainly used for creating navigation menu and to display large amount of data coming from external sources.

14. How to create emails links?

Ans:- To HTML create email links using anchor tags. The code snippet is given below.

```
<a href="mailto:yourmailid.com">Click to open mail link</a>
```

Here as you can see in the href attribute we have passed some addition metadata **mailto:** to make the link work as a mail client link.

15. What is semantic HTML?

Ans:- Semantic HTML is a coding style. It is the use of HTML markup to reinforce the semantics or meaning of the content. For example: In semantic HTML tag is not used for bold statement as well as tag is used for italic. Instead of these we use and tags.

16. What is image map?

Ans:- An image-map is an image with clickable areas.The map element contains a number of <area> elements, that defines the clickable areas in the image map.

```
<img src="planets.gif" width="145" height="126" alt="Planets" usemap="#planetmap">
<map name="planetmap">
    <area shape="rect" coords="0,0,82,126" href="one.htm" alt="One">
    <area shape="circle" coords="90,58,3" href="two.htm" alt="Two">
    <area shape="circle" coords="124,58,8" href="three.htm" alt="Three">
</map>
```

17. How to insert a copyright symbol on a browser page?

Ans:- We can insert a copyright symbol by using © or © in an HTML file.

18. What is a marquee?

Ans:-Marquee is used to put the scrolling text on a web page. You should put the text which you want to scroll within the <marquee>.....</marquee> tag. Code snippet follows.

```
<marqueee>This text is going to scroll in the browser</marquee>
```

19. What is the use of span element ?

Ans:- span element is used to give some special effect to a part of a bigger element. It is an inline element i.e ; it only captures that much space which is required to fit its content. It does not capture the entire width of the browser.

```
<p>Being a programmer <span style = "color: red;">Charles</span> is always busy.</p>
```

And the corresponding out will be:

Being a programmer Charles is always busy.

In the above example Charles will be colored red.

***There are two kinds of tags in HTML according to the width they take by default.
i) Block Elements:-
Block elements capture the entire width of the viewport by default unless we provide them with custom width.

e.g : div, heading elements, paragraph etc

ii)Inline Elements:-

Inline elements , unlike block elements capture only that much width of the viewport as much as they require. They don't capture the entire width of the viewport unless we give them a 100% width.

e.g: span

20. What is the use of iframe tag ?

Ans:- An <iframe> tag is used to display a webpage inside another web page. Like for example if you want to embed a youtube video in your webpage you can easily do it using the iframe tag.

```
<iframe src = "myvideo.com" width = "500px;height =200px"></iframe>
```

21. What is the Quotation element in html ?

Ans:- Quotation element is used to quoting quotations. An example is demonstrated below.

```
<p>Einstein said :-
    <q>Life is like riding a bicycle. To keep your balance you must keep moving.</q>
</p>
```

The above code snippet will yield the following output.

Einstein said :- "Life is like riding a bicycle. To keep your balance you must keep moving."

22. What is comment in html ?

Ans:- Comment is used to make the html code more readable. The browser does not parse the part of the html document written inside comments.

```
<!-- This is how comment is written -->
```

23. What is target attribute inside a html link ?

Ans:- The **target** attribute specifies where to open the linked document. The target attribute can have one of the following values:

_blank - Opens the linked document in a new window or tab
_self - Opens the linked document in the same window/tab as it was clicked (this is default)
_parent - Opens the linked document in the parent frame
_top - Opens the linked document in the full body of the window.

```
<a href="mysite.com" target="_blank">Click to open in a new window</a>
```

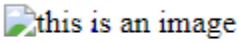
24. What is alt attribute for image in html ?

Ans:- The alt attribute is used to specify some alternative text for the image in case the browser is not able to load it or the src attribute of the image is not found. In this case even if the user is not able to see the image he ca get some idea regarding the image by reading the alt attribute.

The following code snippet shows how to use alt attribute.

```
<img src = "myimage.png" alt = "this is an image">
```

When the browser is not able to fetch the image, it will give an output like below.



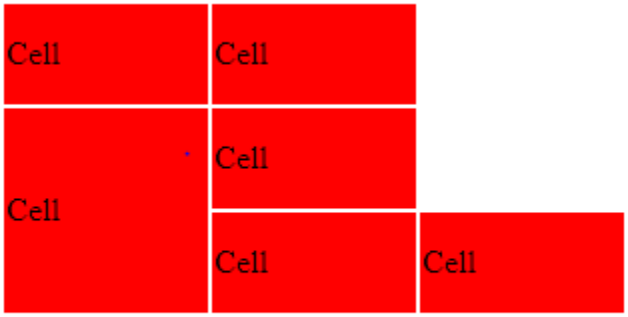
Since the image is not available here, we get to see the text inside the alt text. This is a very valuable attribute as even if the image is not loaded the user can get an idea about what this image is all about.

25. What is rowspan ?

Ans:- rowspan attribute is used to specify how many rows a particular cell will occupy. This can be demonstrated with a simple example as follows.

```
<table>
    <tr>
        <td>Cell</td>
        <td>Cell</td>
    </tr>
    <tr>
        <td rowspan="2">Cell</td>
        <td>Cell</td>
    </tr>
    <tr>
        <td>Cell</td>
        <td>Cell</td>
    </tr>
</table>
```

The following is the output from the browser window.



As we can see the second row has spanned two rows as we have specified the rowspan attribute as 2. This attribute is very useful for case when we try to create asymmetric tables.

1. What is HTML5 ?

Ans:- HTML5 is the latest version of HTML and XHTML with new features like Drawing, Animation, Video and Audio etc. It is used to solve some common structural problems encountered with HTML 4.1. It gives more flexibility to both the web developers, the web designers and enables more exciting and interactive websites in addition to more powerful and efficient applications. HTML5 brings a whole new dimension to the web world. It can embed video on the web-pages without using any special software like Flash. HTML5 is developed in such a way that the developers are not required to waste their time and efforts in creating an error free web page. Firefox, Chrome, Opera, Safari and Internet Explorer all support <!DOCTYPE html>.

2. Why do we use HTML5 ?
Ans:- The main benefit of HTML5 is that it supports Drawing, Animation, Video and Audio. The web developers can decrease the complexity and the time to create applications with animations, play music (audio and video), high quality drawings and other rich content using HTML 5 because it can embed video on the web-pages without using any special software like Flash. HTML5 is far easier for the web designers and the web developers as it tells them how a web page is structured.

3.Why HTML5 specifications came into picture even if there were previous versions ?

Ans:- HTML5 was designed to replace both HTML 4, XHTML, and the HTML DOM Level 2. Major goals of the HTML specification were to:

i) Deliver rich content (graphics, movies, etc.) without the need for additional plugins (e.g., Flash).

i**i)**Provide better semantic support for web page structure through the introduction of new structural element tags.
iii) Provide a stricter parsing standard to simplify error handling, ensure more consistent cross-browser behavior, and simplify backward compatibility with documents written to older standards.
iv) Provide better cross-platform support (i.e., to work well whether running on a PC, Tablet, or Smartphone).

4. How the browser differentiates between HTML5 and HTML 4 ?

Ans:- The browser differentiates between HTML5 and HTML4 by means of the DOCTYPE declaration. In some modern web browsers like Chrome, the browser takes HTML5 by default and all the features of HTML5 will work there. But for some browsers we need to explicitly mention the version of HTML by means of the DOCTYPE to avoid untoward behavior.

5. What are some of the new features added into HTML 5 ?

Ans:- There have been quite some new features added into HTML5. Out of these the most significant are listed below.
i) Canvas
ii) SVG
iii) Audio
iv) Video
v) Geo location
vi) New markup tags
vii) Web storage
viii) Web workers
ix) Google maps
x) Drag n drop
xi) Server sent events etc

6. Which formats HTML 5 video supports ?

Ans:- The video tag supports the following file formats for video.
i) mp4
ii) ogg
iii) webm

7. What is the difference between the progress and meter tags ?

Ans:- The progress tag is used to represent the progress of the task only while the meter tag is used to measure data within a given range.

8. Explain some of the new markup elements in HTML 5 ?

Ans:- Below are some of the new mark ups added into HTML 5.

Tag	Description
<article>	Defines an article in a document
<aside>	Defines content aside from the page content
<bdi>	Isolates a part of text that might be formatted in a different direction from other text outside it
<details>	Defines additional details that the user can view or hide
<dialog>	Defines a dialog box or window
<figcaption>	Defines a caption for a <figure> element
<figure>	Defines self-contained content
<footer>	Defines a footer for a document or section
<header>	Defines a header for a document or section
<main>	Defines the main content of a document
<mark>	Defines marked/highlighted text
<menuitem>	Defines a command/menu item that the user can invoke from a popup menu
<meter>	Defines a scalar measurement within a known range (a gauge)
<nav>	Defines navigation links
<progress>	Represents the progress of a task

9. What is datalist in HTML5 ?

Ans:- The HTML 5 datalist tag provides an auto complete feature on form element. It facilitates users to choose the predefined options.

10. What are web workers ?

Ans:- Web workers give multi threading type functionality to javascript based applications and make the applications more performant. A web worker is a script that runs in the background (i.e., in another thread) without the page needing to wait for it to complete. The user can continue to interact with the page while the web worker runs in the background. Workers utilize thread-like message passing to achieve parallelism.

11. Can a web page contain multiple header and footer tags ?

Ans:- Yes , header and footer tags can be used as many times as you may want. Even individual article and section tags can contain their respective header and footer tags. When the browser renders the tags they will be as per their sequence in the code.

12. Describe the relationship between the header and h1 tags in new HTML5 specifications ?

Ans:- In previous specifications of HTML, only one <h1> element was typically present on a page, used for the heading of the entire page. HTML5 specifies that <h1> represents the top-level heading of a “section”, whether that be the page <body> or an <article> or <section> element. In fact, every <header> element should at least contain an <h1> element. If there is no natural heading for the section, it is a good indication it should not use an <article> or <section> tag.

13. Give an example use of HTML5 video tag ?

Ans:- Below is the example use of html5 video tag.

```
<video width="320" height="240" controls>
    <source src="movie.mp4" type="video/mp4">
    <source src="movie.ogg" type="video/ogg"> Your browser does not support the video tag.
</video>
```

It is a good idea to specify the height and width attribute of the video element as without specified height and width the page may flicker on user interaction. it is a good practice to include two source files so that the browser if supports video will choose the first recognized format.

14. What are the attributes supported by HTML5 video ?

Ans:- Following are the vallrious attributes supported by HTML5 video tags.

Attribute	Value	Description
autoplay	autoplay	Video will start playing automatically.
autobuffer	autobuffer	Video will start buffering automatically.
loop	loop	Video automatically start over again when done.
controls	controls	In order to show the controls.
poster	URL of the image	URL(address) of the image.
src	URL	Address of the video.
width	pixel	Defining the width of the video.
height	pixel	Defining the height of the video.

15. What are the disadvantages of cookies that led to web storage in html5 ?

Ans:- Due to the following disadvantages cookies were replaced by webstorage mechanisms in HTML5.

- Cookies are included with every HTTP request, thereby slowing down your web application by transmitting the same data.
- Cookies are included with every HTTP request, thereby sending data unencrypted over the internet.
- Cookies are limited to about 4 KB of data . Not enough to store required data.

Due to the above mentioned drawbacks web storages came into picture.

16. What are localStorage and sessionStorage. What is the difference between them?

Ans:- With HTML5, web pages can store data locally within the user's browser.
Earlier, this was done with cookies. However, Web Storage is more secure and faster. The data is not included with every server request, but used ONLY when asked for.
The data is stored in name/value pairs, and a web page can only access data stored by itself. Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

The difference between localStorage and sessionStorage involves the lifetime and scope of the storage.
Data stored through localStorage is permanent: it does not expire and remains stored on the user's computer until a web app deletes it or the user asks the browser to delete it.
SessionStorage has the same lifetime as the top-level window or browser tab in which the script that stored it is running. When the window or tab is permanently closed, any data stored through sessionStorage is deleted.

Both forms of storage are scoped to the document origin so that documents with different origins will never share the stored objects. But sessionStorage is also scoped on a per window basis. If a user has two browser tabs displaying documents from the same origin, those two tabs have separate sessionStorage data: the scripts running in one tab cannot read or overwrite the data written by scripts in the other tab, even if both tabs are visiting exactly the same page and are running exactly the same scripts.

17. How to delete localStorage data ?
Ans:- Local storage data has no time limit. To clear a local storage setting you would need to execute the following line of code.

```
<script>
    localStorage.remove('keyName');
</script>
```

where 'keyName' is the key of the value you want to remove.

If you want to clear all settings, you need to execute the following line of code.

```
<script>
    localStorage.clear()
</script>
```

18. What is geolocation API ?

Ans:- HTML5's Geolocation API lets users share their physical location with chosen web sites. JavaScript can capture a user's latitude and longitude and can send it to the back-end web server to enable location-aware features like finding local businesses or showing their location on a map. In broader terms the geo location api returns the geographical location of the user's device.

19. What is Web Forms 2.0?

Ans:- Web Forms 2.0 is an extension to the forms features found in HTML4. Form elements and attributes in HTML5 provide a greater degree of semantic mark-up than HTML4 and remove a great deal of the need for tedious scripting and styling that was required in HTML4.

20. What are server sent events (SSE) ?

Ans:- SSE are special events carrying new information from the server to the client. By SSE a persistent channel is created in one direction (server to browser) so that when ever there is any update in the server it is sent to the client. It eliminates the requirement of continuous polling there by reducing the load .

21. How to incorporate SSE in application ?

Ans:- To use SSE in the application we have to use **eventsource** element inside the document. It should point to the url from which data stream is expected.

22. What are web sockets ?

Ans:- Web Sockets is a next-generation bidirectional communication technology for web applications which operates over a single socket and is exposed via a JavaScript interface in HTML 5 compliant browsers.

Once you get a Web Socket connection with the web server, you can send data from browser to server by calling a send() method, and receive data from server to browser by an **onmessage** event handler.

Following is the API which creates a new WebSocket object.

```
<script>
    var Socket = new WebSocket(url, [protocal] );
</script>
```

Here first argument, url, specifies the URL to which to connect. The second attribute, protocol is optional, and if present, specifies a sub-protocol that the server must support for the connection to be successful.

23. What is Socket.readyState ?

Ans:- The readonly attribute readyState represents the state of the connection. It can have the following values:

- value of 0 indicates that the connection has not yet been established.
- value of 1 indicates that the connection is established and communication is possible.
- value of 2 indicates that the connection is going through the closing handshake.
- value of 3 indicates that the connection has been closed or could not be opened.

24. What is the purpose of Socket.bufferedAmount atribute of WebSocket?

Ans:- The readonly attribute bufferedAmount represents the number of bytes of UTF-8 text that have been queued using send() method.

25. What is <canvas> tag used for ?

Ans:- HTML5 element <canvas> gives you an easy and powerful way to draw graphics using JavaScript. It can be used to draw graphs, make photo compositions or do simple (and not so simple) animations.

26. What are the new form element types in html5 ?

Ans:- There are 10 important new form elements introduced in HTML 5:-

- Color.
- Date
- Datetime-local
- Email
- Time
- Url
- Number
- Telephone

27. Where to use Canvas and SVG

Ans:- Canvas is procedural whereas SVG is declarative. Some reasons to consider SVG instead of canvas are:

i) SVG is scalable, provides the facility of auto scaling icon, logo and chart.
ii) SVG is not supported by the languages whereas canvas elements are manipulated using client-side JavaScript.
iii) DOM handling. It's easy to attach event handlers and manipulate elements like you would for another HTML block. To move an item, you simply change its coordinates but this is not true for a Canvas.

28. What is DataList Tag in HTML5?

Ans:- A <datalist> tag can be used to create a simple auto-complete feature for a webpage. <datalist> is a newly defined HTML tag that comes with the HTML 5 specification. By using this <datalist> tag, we can define a list of data and then we can bind it with an <input> tag. A <datalist> tag specifies a list of predefined options for an <input> element. After binding it, the user will see a drop down list in which all the predefined options will be there for the input. When the user types a character or a string, the user will automatically get the data which depends on the input string or a character. It facilitates the auto complete feature without having to use any external plug-in.

29. What is the use of Drag and Drop in HTML5?

Ans:- Drag and drop is a very common feature and convenient to users. Simply, you need to grab an object and put it at the place you want. This feature is commonly used by many of the online examination websites wherein you have the options to pick up the correct answer,drag it to the answers place holder and drop it.
The Drag and Drop API comes with seven new events to track a drag and drop. The events are dragstart, drag, dragend, dragenter, dragleave, dragover and drop that are triggered during the various stages of the drag and drop operation.

30. What is Audio Tag in HTML 5?

Ans:- This new element allows you to deliver audio files directly through the browser, without the need for any plug-ins. The Audio tag is a new tag introduced in HTML5. You can use it to play audio sound like .mp3, wav, and .ogg. I have used five types of sound formats to show which formats are compatible for the browsers. A WAV file is a common sound format that is supported by most browser versions.

```
<audio controls="controls">
    <source src="URL1" type="audio/mp3" />
    <source src="URL2" type="audio/wma" />
    <source src="URL3" type="audio/x-wav" />
</audio>
```

31. What are the DOM events supported by HTML5 api ?

Ans:- Following are the DOM events supported by HTML5.

onabort	**onblur**	**oncanplay**	**oncanplaythrough**
onchange	onclick	oncontextmenu	ondblclick
ondrag	ondragend	ondragenter	ondragleave
ondragover	ondragstart	ondrop	ondurationchange
onemptied	onended	onerror	onfocus
onformchange	onforminput	oninput	oninvalid
onkeydown	onkeypress	onkeyup	onload
onloadeddata	onloadedmetadata	onloadstart	onmousedown
onmousemove	onmouseout	onmouseover	onmouseup
onmousewheel	onpause	onplay	onplaying
onprogress	onratechange	onreadystatechange	onscroll
onseeked	onseeking	onselect	onshow
onstalled	onsubmit	onsuspend	ontimeupdate
onvolumechange	onwaiting		

32. What are the media elements in HTML 5?

Ans:-
1. Audio
Audio element is used to define or create a music element in your simple HTML page. It supports all the browsers like Internet Explorer 9.0 and above, Chrome, Opera, Firefox and Safari.
This tag defines music or any other audio stream formats.

2. Video
The Video element creates a video element in your HTML page. It supports all the browsers like Internet Explorer 9.0 and above, Chrome, Opera, Firefox and Safari. This tag defines music or any other video stream formats.
3. Track
This element is useful in both the previous elements i.e AUDIO and VIDEO. This element helps to define tracks or we can say simple sectors for the <audio> and <video> elements.
4. Source
Like the track element, the Source element must be used in <audio> and <video> elements to do the control property and structure of the tracks.
5. Embed
It is also called a container because as the name suggests, it is used for defining the containers for the external applications or we can say plug-ins for the Applications.

33. What are the HTML tags which deprecate in HTML5?

Ans:- One of the main points on which HTML5 wins over XHTML2.0 is “backward compatibility”. XHTML2.0 sought to enforce well-written code by using very harsh error handling. If a page returns error based on syntax, the user agent will stop parsing the code. An HTML5 specification states that certain HTML tags should not be used but it is only a guideline to the HTML authors. The implementations, however, must support these tags to be backward compatible.
The tags that are deprecated are the following:

- basefont
- big
- center
- font
- s
- strike
- tt
- u
- frame
- frameset
- noframe
- acronym
- applet
- isindex
- dir

Several tag attributes are also removed. Few of the most notable ones are:

Element	Attribute removed
a,link	rev, charset
img	longdesc, name
html	version
th	abbr
td	scope
all block level elements	align
body	background
img	hspace, vspace
table, tr, th, td	bgcolor
table	border, cell padding, cell spacing
td, th	height, width
table	valign

34. Difference between progress tag and meter tag?

Ans:- A progress tag represents the completion progress of a task whereas the meter tag is used to represent gauges. We can think that a progress tag represents a dynamic data whereas a meter tag represents a static data.
According to the W3C, the meter tag should not be used to indicate progress as to indicate the progress, we have the progress tag.
The meter tag also does not represent a scalar value of an arbitrary range; for example, it would be wrong to use this to report a weight, or height, unless there is a known maximum value.

35. What is the use of Scalable Vector Graphics (SVG) in HTML5?

Ans:- Scalable Vector Graphics (SVG) are the part of the vector-based family of graphics. There are various forms of Raster-based graphics available that stores the color definition for each pixel in an array of data. Some of the most common Raster-based formats used on the web today are JPEG (Joint Photographic Experts Group), GIF (Graphics Interchanged Format) and PNG (Portable Network Graphics). SVG is a language for to describe 2D vector graphics in XML.Creation of an SVG image is a very different process. To create any other Raster images like JPEG, PNG or GIF, we use image editing software like Photoshop and so on but SVG images are XML based file so they can be created in any other text editor. There is a tool also available (inkspace). By using this tool, you can draw and create SVG images very conveniently.

You can use SVG XML tag to create shapes. The following table shows different types of shapes that can be created using SVG.

Element	Description
line	Creates Simple line
circle	Creates Circle
rect	Creates Rectangle
ellipse	Creates Ellipse
polygon	Creates Polygon
polyline	Creates Multiline Shape
path	Creates Arbitrary Path
text	Allows to Creates Text

36. What is the use of cite tag in HTML5?

Ans:- The <cite> tag indicates a citation. It represents the title of a work (e.g. a book, paper, essay, poem, score, song, script, film, TV show, game, painting, sculpture , play , exhibition , etc.).
The <cite> tag is an inline tag that indicates "defining a citation". The text within the <cite> tag is shown in Italics. The cite tag must have a start and end tag.In this tag the "title" attribute defines the title of the Text within the <cite></cite> tags.In HTML5 , the <cite> tag defines the cited title of a work whereas HTML 4.01 implied that the <cite> tag can be used to indicate the name of a person.
An example of cite tag is given below.

```
<cite title="value">Some Text Here</cite>
```

And the corresponding output will be as below.

Some Text Here

As we can see that the output is some sort of formatted to make it look like a citation.

37. What are Waves in HTML?

Ans:- A sine wave is a mathematical function that is repeats at a regular interval of time. The function is used in many fields including mathematics, physics, and engineering. We can also say that a sine wave is a smooth wave.
It has the following properties:

- The sine wave is blue whenever the value is positive.
- The sine wave is red whenever the value is red.
- The thickness of the line is directly proportional to the absolute value of the magnitude of the wave. For example, where the sine value reaches 0, the wave is absent. On the X-axis, we will map the angle Theta. Theta will vary from 0 degree to 1040 degrees. On the Y-axis, we will map the sin (Theta). For this, we will use the Math function Math.sin. The Math.sin function takes angles in radians. So the angle is first multiplied by PI / 180.

38. What is Web SQL Database in HTML 5?

Ans:- Web SQL is a very interesting feature, even though it isn't part of the HTML 5 specification but it is a separate specification and it can still help to develop Web Applications. Web SQL is used to manipulate a client-side database. Since I am saying that it is good to use, there is a disclaimer for its use; it is risky because it stores data at the client side, not on the server side. So always remember, don't store information sensitive to the server inside it.

Core Methods of Web SQL:
The following are the 3 core methods of Web SQL that are most important:

i) openDatabase
ii) transaction
iii) executeSql

Creating and Opening Databases:
Using the openDatabase method, you can create an object for the database. If the database doesn't exist then it will be created and an object for that database will be created. We also don't need to worry about closing the connection with the database.To create and open the database, we need to use the following code snippet.

```
var dbObj = OpenDatabase('[Database_Name]', '[Version_Number]', '[Text_Description]',
'[size]', '[Creation_Callback]');
```

39. What is HTML5 Contenteditable Attribute?

Ans:- One of the new features in HTML 5 is the contenteditable attribute. In HTML 5, any element can be editable. By using some JavaScript event handler, you can transform your web page into a fast rich text-box. This feature is mainly applied in Content Management Systems. By using this, you can edit content directly from the browser.The contenteditable attribute is an enumerated attribute whose keywords are the empty string, true and false. The empty string and the true keyword equates to the true state. The false keyword implies the false state. In addition, there is a third state, the inherit state, which is the missing value default (and the invalid value default).

States of content editable attribute:
According to WHATWG.org, there are the following 3 states of the contenteditable attribute:

State	Description	How to write?
true	Indicates that element is editable	contenteditable=" " / contenteditable="true"
false	Indicates that element is not editable	contenteditable="false"
inherit	ndicates that the element will be editable if and only if, its immediate parent element is editable	contenteditable="inherit"

40. What is Vibration API in HTML5?

Ans:- Vibration is a simple, a nice way of alert when you get a new message or a phone call. It is especially useful when you are in a noisy environment or the place where you feel the ringing would be a distraction to others. It is interesting to know that HTML5 is now providing us to play with the vibration on the devices but the HTML5 Vibrate API supports only the recent version of Firefox and Chrome.

41. What is the Battery Status API in HTML5?

Ans:- When a users downloads an application for their devices, they are more conscious of the battery usage of the application. So as a mobile application developer, you should consider the battery usage of your Application.If you are developing a Web Application for a mobile device then your choice is to use HTML5's Battery Status API, if you are concerned about the user's device battery status/charging levels. Yes, HTML5 provides an API for a device's battery.W3.org says: "The Battery Status API specification defines a means for the web developers to programatically to determine the battery status of the hosting device".

We can check the availibility of battery status by the following syntax.

```
<script>

    var battery = navigator.battery || navigator.webkitBattery || navigator.mozBattery ||
        navigator.msBattery;
    if (battery) {
        //your code if battery is available
    }
</script>
```

There are four basic properties available in the battery status API.

1. Charging: Charging is a type of Boolean and a read only that indicates whether the device is charging the battery. The default value is true.

2. ChargingTime: ChargingTime is type is double and a read only that gives you the remaining time in seconds to charge the device battery fully. The default value is 0.

3. DischargingTime: DischargingTime is the type of double and read only that represents the remaining time for a complete discharge of the device battery. The default value is calculated based on the other property values.

4. Level: Level is a type of double and read only that represents the battery level in the scale of 0 - 1.0. The default value is 1.0.

42. What are Web Workers APIs in HTML 5?

Ans:- Web Workers APIs provide a way in JavaScript to run something in the background that can do tasks without interfering with the user interface. As per the W3C standard "It is a JavaScript script executed from an HTML page that runs in the background, independently of other user-interface scripts that may also have been executed from the same HTML page.
Web workers are able to utilize multi-core CPUs more effectively."

There are two kinds of web workers.

1.Dedicated workers:

- A dedicated worker is accessible from the parent script that created it
- Wide browser support: All
- It is simply tied to its main connection

2.Shared workers:

- A shared worker can be accessed from any script of the same origin.
- Limited browser support: Chrome 4.0+, Safari 5.0+ and Opera 10.6+.
- It can work with multiple connections.

Basically Web Workers work in the following three steps:
i) First it should be executed on separate threads.
ii) It should be hosted in separate files from the main page.
iii) Finally a Worker object needs to be instantiated to call them.

43. Describe Progress Bar in HTML 5?

Ans:- Sometimes a task is running within a program that might take a while to complete. A user friendly program provides some information to the user that the task is happening. It also tells how long the task might take and how much the task has been done or completed. One of the best way to show all of these activities is with the Progress Bar control.
A progress bar in its simplest format looks like this.

```
<progress></progress>
```

The progress bar has mainly two possible attributes.

1.max : specifies the maximum work required to complete the work
2.value: specifies how much work has been completed

44. What is Microdata in HTML 5?

Ans:- Microdata is used to nest metadata within an existing content on the web pages. This mechanism allows machine-readable data to be embedded in HTML documents in an easy to write manner, with an unambiguous parsing model. Microdata allows us to define our own customized elements and start embedding custom properties in our web pages. It's purpose is not to make a new widget appear on our web page but to help automated programs like Google understand and better handle the content of our web pages. A Microdata consists of a group of name-value pairs. The group of name-value pairs is called items, each name : value property is called as a property and properties are represented by regular elements. Microdata defines five Global HTML attributes that can be applied to any HTML5 tag.

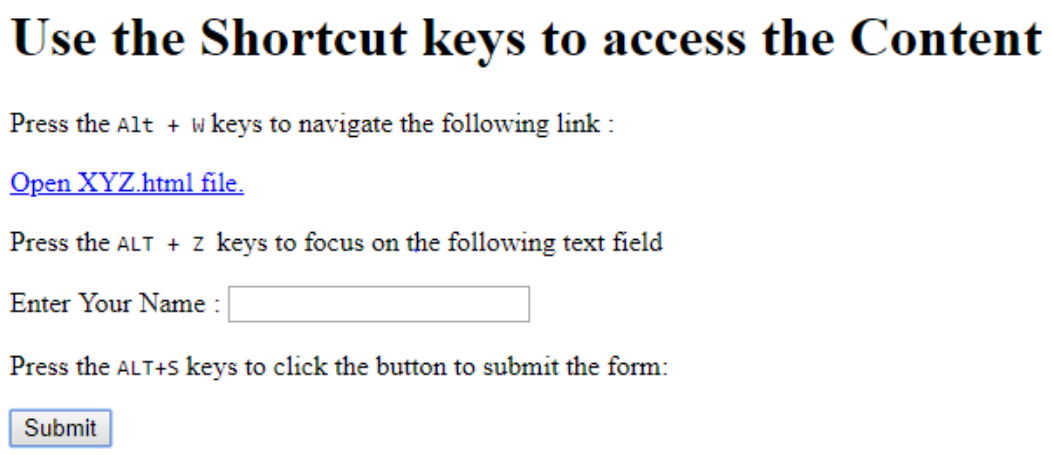
45. What is keyboard shortcut in HTML 5?

Ans:- For Displaying the Keyboard text, we can also create the keyboard shortcut to perform various operations such as clicking a link or a button. We can use the accesskey attribute when defining the element to provide the keyboard shortcut to that element or control. Let's create a Web Page, named "accesskey.html" and understand 'How to create a keyboard shortcut.

The following example demonstrates this functionality.

```
<body>
    <h1>
        Use the Shortcut keys to access the Content
    </h1>
    <p>
        Press the
        <kbd>Alt + W</kbd> keys to navigate the following link :
    </p>
    <a href="XYZ.html" accesskey="w" target="">Open XYZ.html file. </a>
    <p>
        Press the
        <kbd>ALT + Z </kbd>keys to focus on the following text field
    </p>
    Enter Your Name :
    <input type="text" name="name" accesskey="z">
    <p>
        Press the
        <kbd>ALT+S</kbd> keys to click the button to submit the form:</p>
    <input type="submit" accesskey="s" onclick="alert('Form submitted successfully.')">
</body>
```

And the corresponding output as displayed in the web browser will be as follows.



Now When we press ALT + * Key:

ALT + W : The focus goes on the hyperlink and we areredirected to the page specified by the hyperlink (XYZ.html)
ALT+Z : When we press ALT+Z key then get a focus on the Enter your Name text field.
ALT+S: This focuses on Submit button and give an alert box.
This is how the keyboard short cut in HTML5 specification work.

1. What is CSS ?

Ans:- CSS stands for **C**ascading **S**tyle **S**heet. It is used to style the mark up provided by HTML. So the name stylesheet is justified. Cascading means series connection. In CSS the styles are connected in series connection one after another just like electrical series connections, separated by semicolon. Thus the name Cascading stylesheet is justified.

2. What is the main role of CSS in a web page ?

Ans:- HTML provides the basic mark up of a web page. But it is way too boring with all black and white text. Here comes css into the picture by making the mark up fill with color and styles. Without CSS or its pre-processors the WWW would not have been a so beautiful place.

3. What is the origin of CSS ?

Ans:- SGML (Standard Generalized Markup Language) is the origin of CSS.

4. What are the versions of CSS ?

Ans:- CSS has undergone the following major versions from its inception.

- CSS1
- CSS2
- CSS2.1
- CSS3
- CSS4

5. What are the advantages of CSS ?

Ans:- Following are some of the advantages of Cascading Style sheets.

- Site wide consistency
- Page reformatting
- Bandwidth
- Accessibility
- Content separated from presentation

6. How many ways are there to integrate CSS in a web page ?

Ans:- There are three ways by which we can integrate CSS into a web page.

1. Internal
2. Inline
3. External

7. What is the syntax of CSS ?

Ans:- The basic syntax of CSS is as follows.

```
selector {
        property1: value1;
        property2: value2;
    }
```

Here **selector** is the selector of the element(name, class, id or attribute) for which style is to be applied.

8. How to write inline CSS ?

Ans:- The CSS which is written inside the html element itself is called inline CSS. This method of writing CSS is not recommended though it is used to over ride other style methods. Its precedence is superior than class and id selector methods , but inferior than styles described as **!important**.
A simple example of writing inline CSS is illustrated below.

```
<html>
    <head></head>
    <body>
      <h1 style = "color: red;border:solid 3px  green;width:50%;">This is a heading with red color and a border</h1
    </body>
</html>
```

And the output of the above code when rendered by the browser will be as follows.

This is a heading with red color and a border

9. How to write internal CSS ?

Ans:- Internal CSS is written inside the <head> or <body> section of the document. The CSS written in this method is applicable to the current document only.
This style of writing styles has the advantage that it will not affect any other document. But this way is not ideal for cases where so many styles are to be written as that will make the page lengthy and its accessibility will also be hampered.

```
<html>
    <head>
        <style>
            h1 {
                color: blue;
                border: 2px solid red;
                width:50%;
            }
        </style>
    </head>
    <body>
      <h1 >This is a heading with red color and a border</h1>
    </body>
</html>
```

And the resulting output of the above code will look like this.

10. How to refer external stylesheet ?

Ans:- As the name suggests this method of writing CSS involves an external CSS file which is to be referenced from the current document.The CSS file styles are then available in all the documents where it is been referenced. It is the recommended method of writing CSS. The external CSS file is referenced in the head section of the document and the syntax to reference is as follows.

```
<link rel = "stylesheet" type = "text/css" href = "style.css" >
```

Here the **rel** attribute specifies the relation of the html file with the stylesheet, the **type** attribute specifies the type of the file which we intend to reference from the current html file and the **href** (hyper reference) specifies the pth to the css file**.** The path is given relative to the file from which we want to reference the CSS file.

11. What is the advantage of external stylesheet ?

Ans:- Following are the advantages of using external stylesheets.

- You can create classes for reusing it in many documents.
- By using it, you can control the styles of multiple documents from one file.
- In complex situations, you can use selectors and grouping methods to apply styles.

12. What is a ruleset ?

Ans:- Ruleset is used to identify that selectors can be attached with other selectors. It has two parts namely

- Selector
- Declaration

13. What is the difference between class selectors and ID selectors ?

Ans:- By class selectors multiple elements with the same class can be selected while for id selectors only one unique element can be selected as with w3 standardization same id can not be assigned to multiple elements.

14. What is RWD ?

Ans:- RWD stands for Responsive Web Design. This technique is used to display the designed page perfectly on every screen size and device. For example: Mobile, Tablet, desktop, laptop etc. You don't need to create a different page for each device.

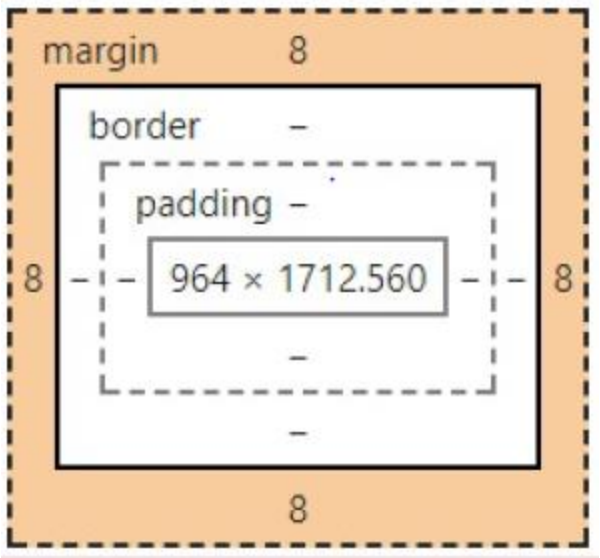
15. What are css sprites and why do we use them ?

Ans:- If a web page has large no. of images that takes a longer time to load because each image separately sends out an http request. The concept of CSS sprites is used to reduce the loading time for a web page because it combines the various small images into one image. It reduces the number of http requests and hence the loading time.

16. What is CSS box model ?

Ans:- With the latest css specifications every element in the document is considered a set of nested rectangular boxes .The CSS box model is used to define the design and layout of elements rendered inside the browser.
The properties which are represented by the rectangles are listed below.

- Margin
- Border
- Padding
- Content



Here the inner most rectangle is the content of the element and it is nested inside by levels of padding, border and margin respectively.

17. What is css float property ?

Ans:- The css float property as the name suggests is used to make the element float inside the document. By this property we can make the elements float left or right or center according to the requirement. A very practical use of the float property is to float the list items inside the document to create a horizontal navigation menu.

18. What is z-index ?

Ans:- 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.
Z-Index can take the following values:

- **Auto:** Sets the stack order equal to its parents.
- **Number:** Orders the stack order.
- **Initial:** Sets this property to its default value (0).
- **Inherit**: Inherits the property from its parent element.

19. What is the difference between visibility hidden and display none properties ?

Ans:-
visibility: hidden: simply hides the element but it will occupy space and affect the layout of the document.
display: none: also hides the element but will not occupy space. The elements after the current element will get elevated to occupy the space cleared by display:none.

20. What is meant by tweening ?

Ans:- It is the process of generating intermediate frames between two images.
It gives the impression that the first image has smoothly evolved into the second one.
It is an important method used in all types of animations.
In CSS3, Transforms (matrix, translate, rotate, scale etc.) module can be used to achieve tweening.

21. What are css psuedo classes ?

Ans:- A pseudo-class is used to define a special state of an element using CSS.
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

The syntax to use psuedo class is as follows.

```
selector:pseudo-class {
    property:value;
}
```

There are mainly 4 types of psuedo classes which are listed below.

- link: when an element or link is not clicked/unvisited
- visited : when the element is clicked or the link is visited
- hover : applied when element gets a mouse hover
- active : applied when the link is active now

```
/* unvisited link */
a:link {
    color: red;
}
/* visited link */
a:visited {
    color: green;
}
/* mouse over link */
a:hover {
    color: blue;
}
/* selected link */
a:active {
    color: black;
}
```

22. What css property is used to remove the bullet marks from the list items ?

Ans:- The following CSS style can be used to remove the bullet marks from a list item.

```
<style>
    li {
        list-style-type: none;
    }
</style>
```

23. How to remove the default underline coming with the html anchor tag ?

Ans:- To remove the underline from an anchor tag we can use text-decoration property. The text-decoration property can be used as follows.

```
<style>
    a {
        text-decoration: none;
    }
</style>
```

The text-decoration property has three possible values which are used most.

- underline : creates an underline
- overline : creates a line over the element
- line-through : creates a line that passes through the element

24. What are the various media types used in css ?

Ans:- Different media has different properties as they are case insensitive.
Different media types used in CSS are listed below. For creating responsive web pages we need a very clear understanding of the dimensions of each type of device.

- Aural – for sound synthesizers and speech
- Print – gives a preview of the content when printed
- Projection- projects the CSS on projectors.
- Handheld- uses handheld devices.
- Screen- computers and laptop screens.

25. What are contextual selectors ?

Ans:- By contextual selectors we can select special occurrence of elements so that elements with the same tag name or class name will not be affected.
In the below example we will change the color of the inside paragraph to red but the color of the external paragraph color will not be altered.

```
<!DOCTYPE HTML>
<html>

<head>
    <style>
        div p {
            color: red;
            font-size: 30px;
        }
    </style>
</head>

<body>
    <p>Hi! I am the external paragraph.</p>
    <div>
        <p>Hi! I am an internal paragraph.</p>
    </div>
</body>

</html>
```

And the corresponding output as seen in the browser will be as like below.

Hi! I am the external paragraph.

Hi! I am an internal paragraph.

As we can see in the output only the color and font-size of the internal paragraph are affected by our styles whereas the external paragraph is not effected. In the code snippet we can see that we are only targeting paragraph(s) which are children of the div element. So the paragraphs which are outside of the div element are not affected.

26. What are the properties to define the dimensions of elements in css ?

Ans:- Dimension properties in any element can be defined by the following CSS properties.

- Height
- Max-height
- Max-width
- Min-height
- Min-width
- Width

27. What is graceful degradation ?

Ans:- In case the browser fails to load certain styles of an element, it will continue to work properly in the presence of a graceful degradation. The latest browser application is used when a webpage is designed. As it is not available to everyone, there is a basic functionality, which enables its use to a wider audience. The best example in this regard is the case when the image is unavailable for viewing, text is shown with the alt tag.

28. What is progressive enhancement ?

Ans:- It's an alternative to graceful degradation, which concentrates on the matter of the web. The functionality is same, but it provides an extra edge to users having the latest bandwidth. It has been into prominent use recently with mobile internet connections expanding their base.

29. How css code can be commented ?

Ans:- CSS code can be commented like the following example. Here we can see that one property is commented while the rest are active. We use **/* ... */** to comment code. Any code in between the **/*** block will be commented and will not be treated as executable code by the browser. If you are writing code with Visual Studio Code editor then , there is a keyboard shortcut to comment code. Select the code which you want to comment and use the shortcut **ctrl** + **/ .**

```
<style>
    p {
        color: red;
        border: 2px solid green;
        /* font-size: 30px;
        font-weight: bold; */
    }
</style>
```

Here the font-size and font-weight properties are commented.

30. What is @ rule ?

Ans:- Rule, which is applicable in the entire sheet and not partly, is known as at-rule. It is preceded by @ followed by A-Z, a-z or 0-9.

31. How to select all paragraphs with a lang attribute ?

Ans:- We can select all paragraphs with a lang attribute by the CSS attribute selector. The attribute selector is a type of selector in CSS along with class, element and id selectors which selects a particular element from its attribute value. We can use the attribute selector like the following example.
p[lang] : Selects all paragraph elements with a lang attribute.

32. How css style over riding works ?
Ans:- Following is the rule to override any Style Sheet Rule −

- Any inline style sheet takes highest priority. So, it will override any rule defined in <style>...</style> tags or rules defined in any external style sheet file.
- Any rule defined in <style>...</style> tags will override rules defined in any external style sheet file.
- Any rule defined in external style sheet file takes lowest priority, and rules defined in this file will be applied only when above two rules are not applicable.

33. In how many ways css color can be given ?

Ans:- We can specify the color values in five various formats. Following table lists all the formats which we can use to color elements.

Format	Syntax	Example
Hex Code	#RRGGBB	p{color:#FF0000;}
Short Hex Code	#RGB	p{color:#6A7;}
RGB %	rgb(rrr%,ggg%,bbb%)	p{color:rgb(50%,50%,50%);}
RGB Absolute	rgb(rrr,ggg,bbb)	p{color:rgb(0,0,255);}
keyword	aqua, black, etc.	p{color:teal;}

34. What are browser safe colors ?

Ans:- There is the list of 216 colors which are supposed to be most safe and computer independent colors. These colors vary from hexa code 000000 to FFFFFF. These colors are safe to use because they ensure that all computers would display the colors correctly when running a 256 color palette.

35. What are the css background properties ?

- background-image
- background-repeat
- background-attachment
- background-position
- background-color

36. How to set a background image ?

Ans:- We can set the background image of an element in the following manner.

```
<style>
    body {
        background-image: url("mybg.jpg");
    }
</style>
```

37. How to repeat the background image in horizontal direction only ?

Ans:- To repeat the background-image in the horizontal direction we have to use the **background-repeat** CSS property. The following example demonstrates this concept.

```
<style>
    body {
        background-image: url("gradient_bg.png");
        background-repeat: repeat-x;
    }
</style>
```

Here as we can see we have used the background-repeat property with the value repeat-x, which tells the browser to repeat the background image in the horizontal direction.

The output of the above code snippet as captured in the browser will be as follows.

38. Which property is used to position the background image ?

Ans:- Background-position property is used to position the background image. The code use of this property is shown below.

```
<style>
    body {
        background-position: right top;
    }
</style>
```

This property has preferably two values. The first value specifies the horizontal position from where the background image will start and the second value specifies the vertical positioning of the background image. Another use case of this property is given below.

```
<style>
    body {
        background-position: 10% 20%;
    }
</style>
```

39. What is the background shorthand syntax ?

Ans:- To shorten the code it is possible to write all the background properties in one single line. The sequence of properties for the short hand notation is as follows.

background-color => background-image => background-repeat =>background-attachment =>background-position
The following example demonstrates the use of these shorthand properties.

```
<style>
    body {
        background: #ffff00 url("myimg.png") no-repeat right top;
    }
</style>
```

Here in the above example we can see different space separated properties getting used to specify corresponding values.

40. What are the border styles available with current css3 specification ?

Ans:- The **border-style** property specifies what kind of border to display for the concerned element. The following values are allowed against this CSS property.

- dotted - Defines a dotted border
- dashed - Defines a dashed border
- solid - Defines a solid border
- double - Defines a double border
- groove - Defines a 3D grooved border. The effect depends on the border-color value
- ridge - Defines a 3D ridged border. The effect depends on the border-color value
- inset - Defines a 3D inset border. The effect depends on the border-color value
- outset - Defines a 3D outset border. The effect depends on the border-color value
- none - Defines no border
- hidden - Defines a hidden border

41. What is the border shorthand property ?

Ans:- The border shorthand property specifies all the border properties in one statement. The properties include

- border-width - specifies the width of the border (pixels)
- border-style - specifies the style of the border(dashed, solid etc)
- border-color - color of the border

An example of border shorthand property is given below.

```
<style>
    p {
        border: 2px solid red;
    }
</style>
```

42. How to give different border-width to different sides of an element ?

Ans:- There is a shorthand by which the border width for all the sides can be given differently. Here we give the widths of different sides in clockwise manner. Here the values correspond to border top ,right,bottom and left.(clockwise rotation).
An example is illustrated below.

```
<style>
    p {
        border-width : 2px 5px 7px 10px;
    }
</style>
```

43. What is margin ?

Ans:- The CSS margin properties are used to create space around elements, outside of any defined borders.With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

There are primarily 4 types of margins available.

- margin-top
- margin-right
- margin-bottom
- margin-left

44. Give an example of css margin property ?

Ans:- The following example demonstrates different types of margin properties available that can be applied to elements.

```
<style>
    p {
        margin-top: 100px;
        margin-bottom: 100px;
        margin-right: 150px;
        margin-left: 80px;
    }
</style>
```

45. Describe the margin shorthand property ?

Ans:- The margin property is a shorthand property for the following individual margin properties. There are four values assigned to the margin short-hand property which are given in clock wise manner.

- margin-top
- margin-right
- margin-bottom
- margin-left

```
<style>
    p {
        margin: 25px 50px 75px 100px;
    }
</style>
```

46. How to place a div at horizontally center position ?

Ans:- We can position an element exactly at the horizontal center of its parent element with the margin property in the following manner.

```
<style>
    .inner_div {
        margin: 0 auto;
    }
</style>
```

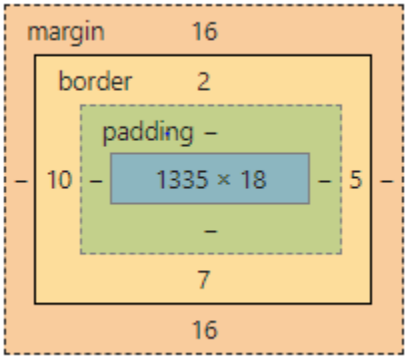
47. What does it mean to have a margin inherit value ?

Ans:- If we want the element to inherit the margin value from its parent element then we set the margin value to be inherit. By this property the margin of the concerned element will be same as the margin value of its parent.

```
<style>
    div {
        margin: inherit;
    }
</style>
```

48. What is css padding property ?

Ans:- The CSS padding properties are used to generate space around an element's content, inside of any defined borders.
In the CSS box model, padding is defined by the green rectangle.



49. What is padding shorthand property ?

Ans:- The css padding short hand property has four values in sequence.

- padding-top
- padding-right
- padding-bottom
- padding-left

An example of the padding short hand property is given below.

```
<style>
    div {
        padding: 5px 10px 10px 5px;
    }
</style>
```

50. How to handle css padding property with the element width ?

Ans:- By box-sizing: border-box property the css padding property is used along with the width property so that with increase in padding the width f the element gets adjusted to the maximum value specified to it.

51. What does *div,p* selector select ?

Ans:- The selector div,p selects all the div(division) and the p(paragraph) elements inside the document.

52. What does div p selector select ?

Ans:- Selects all the p elements that are anywhere nested(child elements) inside the div element.

53. What does div > p selector select ?

Ans:- Selects all the p elements that have an immediate parent a div element.

```
<div>
    <p>This paragraph will be selected with div > p selector</p>
    <div>
        <p>This paragraph will not be selected </p>
    </div>
</div>
```

54. What does div+p selector select ?

Ans:- Selects all the p element that are paced immediately after the div element.

```
<div>
    <p>This paragraph will not be selected</p>
</div>
<p>This paragraph will be selected</p>
```

55. What does div~p selector select ?

Ans:- It selects all p elements that are immediately preceded by the div element.

56. How to select all the anchor tags whose href attribute starts with "https" ?

Ans:- To select all anchor tags with attribute starting with https we need to do the following.

```
<style>
    a [href^="https"] {
        color: red;
    }
</style>
```

57. How to use grouping in css ?

Ans:- Grouping is mainly used for applying CSS style for multiple HTML elements and this can be done with single declaration. Example given below is the example of the grouping.

```
<style>
    h2,
    h3 {
        color: red;
    }
</style>
```

58. What are the font properties available ?

Ans:- Below are the list of font attributes in CSS3 :-

- Font-Variant
- Font-Family
- Caption
- Font-Style
- Font-Size
- Icon

59. What are the important new features in CSS3 ?

Ans:- Below are the some of the new features added in CSS3.

- Border Images
- New Web fonts
- Multi Column layout
- Box Shadows
- Text Shadows
- Transform property

60. What are the available values for position property ?

Ans:- Below are the list of possible values for attribute – “Position” -

- Static
- Inherit
- Fixed
- Absolute
- Relative

61. What are the new background properties added in CSS3 ?

Ans:- Below are the new background properties are added in CSS3-

- background-origin
- background-clip
- background-size

62. What is word-wraping ?

Ans:- By word-wrapping we ban break bigger words to the next line. We can use the word-wrap property in the following manner.

```
<style>
    .el {
        word-wrap: break-word;
    }
</style>
```

63. What are the new text properties in CSS3 ?

Ans:- Below are the list of text properties added in CSS3 –

- Word-wrap
- Text-overflow
- Word- break

64. What are the properties of CSS3 transition ?

Ans:- Below are the properties for transition in CSS3 –

- Transition-delay
- transition-property
- transition-duration
- transition-timing-function

65. What are the types of CSS3 gradients?

Ans:- Below are the different types of gradients in CSS3 –

- Radial gradients
- Linear gradients

66. What is the difference between "cell-padding" and "cell-spacing"?

Ans:- The difference between cell-padding and cell-spacing are as follows.

- Cell-Padding – This used to leave the space between the content of cell and wall/border of the cell.
- Cell-Spacing – This used to specify the space between the cells.

67. What are CSS3 media queries and why they are used ?

Ans:- CSS media queries are the filters that make responsive web design (RWD) possible. With the introduction of laptops, tablets, and smartphones with screens of varying sizes and aspect ratios, RWD is crucial for modern day app development. CSS media queries adjust content style based on device characteristics like width, height, orientation, resolution, and display type. When used properly, the end result is a website or app capable of providing a sleek UI/UX across multiple devices.

68. What is a CSS preprocessor ?

Ans:- A preprocessor is an abstraction layer built on top of CSS. Preprocessors extend the functionality of CSS by offering powerful features like **variables**, inheritable **classes** called extends, and "function-like" **mixins**. Sass, LESS, and Stylus are some of the more well known preprocessors.

69. What are CSS vendor prefixes ?

Ans:- Vendor prefixes are extensions to CSS standards that can be added to these features to prevent incompatibilities from arising when the standard is extended. CSS vendor prefixes for some common platforms are listed below.

- -moz-: Mozilla Firefox
- -ms-: Internet Explorer
- -o-: Opera
- -webkit-: Android, Chrome, iOS, and Safari

70. Why CSS is so fast ?

Ans:- CSS acts very fast while filling the styles of elements because it works in the system hardware level.

1. What is less ?

Ans:- Less is a css pre-processor that gets compiled into valid CSS. Less is a dynamic style sheet producing language that can be compiled into Cascading Style Sheet (CSS) and runs on client side and server side both.

2. How to create a less file ?

Ans:- Creating and Storing Less file is same as creating and storing CSS files. You can create a Less file with a .less extension or you can rename an existing .css file to .less file. You can also write Less code with existing CSS code.

3. How many ways LESS can be used ?

Ans:- Less can be used in following three ways:

- Via npm LESS can be used on the command line.
- Download as a script file for the browser.
- It can be used for third party tools.

4. What are the advantages LESS provides on top of CSS ?

Ans:- Less file is compiled into plain CSS file but it introduces a taste of dynamism which CSS lags. Less achieves this dynamism by means variables, mixins and functions.

5. How are variables represented in Less ?

Ans:- Variables are represented with a @sign like **@myColor: red;**

6. What are mixins in Less ?

Ans:- Mixins enable embedding all the properties of a class into another class by including the class name as one of its properties. It works like variables but for whole class.

7.How nesting is achieved in Less ?

Ans:- Nesting in LESS programming specifies the clustering of statements inside other statements. It forms a group of related codes. For example, if you add a code snippet and add another code inside it, then the code snippet is called nesting.

8. What are the color channel functions used in Less ?

Ans:- List of the color channel functions used in LESS is as follows.

- hue
- saturation
- hsvhue
- saturation
- hswalue

- red
- green
- blue
- alpha
- luma
- luminance

9. What is data URI in Less ?

Ans:- Data URI is a technique that allows developers to avoid external image referencing and instead embed them directly into a stylesheet. Data URI is the best technique to reduce HTTP requests.

10. What is the e() function in Less ?

Ans:- The e() function is used to escape a value so that it passes straight through to the compiled CSS, without being noticed by the LESS compiler.

11. What is the use of escaping ?

Ans:- Following are the use of escaping in LESS:

- When you need to output CSS that is not valid CSS syntax
- Proprietary syntax not recognized by LESS
- LESS compiler will throw an error if not used
- Simple prefix with ~ symbol and put in quotes

1. What is Javascript ?

Ans:- JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages. Javascript gives life to the otherwise dead HTML pages so that they can respond to user interaction.

2. Why javascript is so popular ?

Ans:- Due to the following reasons JavaScript is a popular choice in software development.

- javascript is the only language that is fit to run in the browser
- javascript comes bundled with the browser
- no need for any extra library or package to run javascript environment
- very light-weight and easy to learn
- it's the only programming language that can run in both front end and back-end (nodejs)

3. What are the main advantages of using javascript ?

Ans:- Following are some of the advantages of using JavaScript −

- Less server interaction −You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.

- Immediate feedback to the visitors −They don't have to wait for a page reload to see if they have forgotten to enter something.

- Increased interactivity −You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.

- Richer interfaces −You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

4. What is the difference between JavaScript and jscript ?

Ans:- Netscape provided the JavaScript language. Microsoft changed the name and called it JScript to avoid the trademark issue. In other words, you can say JScript is same as JavaScript, but it is provided by Microsoft.

5. What is the difference between undefined and 'not defined' in JavaScript?

Ans:- To understand this answer we have to understand the difference between declaration ad definition. This is a very important concept of hoisting.
Declaring a variable means we create a memory space for that variable. Defining a variable means we are assigning some value to the variable.

```
<script>
    var x ; // variable declaration
    x = 10; // variable definition
</script>
```

Now take the below example where the variable is declared and defined at the same time.

```
<script>
    var x ; // variable declaration
    x = 10; // variable definition
    var y = 10; // both declaration and definition
</script>
```

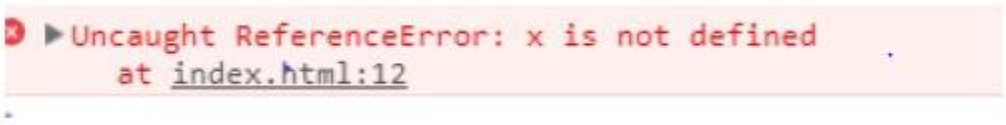
Consider a case where we want to access the variable before declaring it like the below example.

```
<script>
    console.log(y);
    var y = 10;
</script>
```

Here the output will be undefined as during the hoisting context the browser will create a memory space for the variable first and assign it a value equal to undefined. So when we try to access that variable before it is actually defined, the browser has already assigned it **undefined**.
Now what if the variable is not there at all , like in the below example.

```
<script>
    console.log(y);
    console.log(x);
    var y = 10;
</script>
```

Here variable **x** not present in the hoisting context (when the browser takes up variables and functions and keep them in a stack). The naturally the output will be an error saying **x is not defined** as shown in the browser console.



So this is basically the difference between undefined and is not defined.

6. What is BOM ?

Ans:- The **browser object model** (BOM) is a hierarchy of **browser objects** that are used to manipulate methods and properties associated with the Web browser itself. Objects that make up the BOM include the window object , navigator **object** , screen **object** , history, location object , and the document object**.** Through the BOM user interacts with the browser.

7. What is DOM ?

Ans:- DOM stands for *Document Object Model*. A document object represent the html document. It can be used to access and change the content of html.
Document Object Model is the object where the user can have access to the html nodes. For example if we want to select the **body** node from the document we can access it like below.

```
<script>
    var x = document.getElementsByTagName("BODY")[0];
</script>
```

8. What is window object ?

Ans:- The **window object** represents a window in browser. An object of window is created automatically by the browser.
The window object has methods like alert(), setTimeout() etc. Whenever the execution context is inside these functions the current scope always points to the window object.

9. What is the history object ?

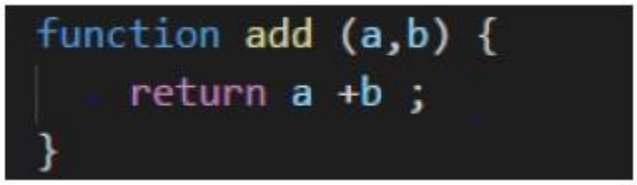
Ans:- The history object is used to store the locations of past pages which are part of the current history. It has basically three methods.

- history.back()
- history.forward()
- history.go(number)

10. How many ways are there to write functions in javascript ?

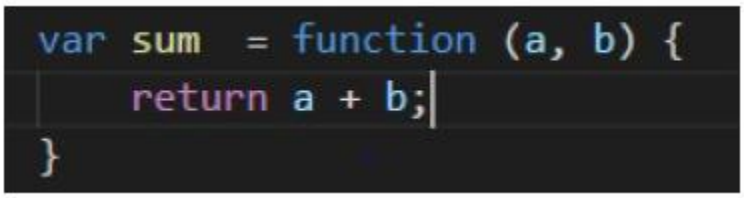
Ans:- Functions in JavaScript can be written in two ways.

- Function declaration : An example of function declaration is given below.



```
function add (a,b) {
    return a +b ;
}
```

- Function expression : An example of function expression is given below.



```
var sum  = function (a, b) {
    return a + b;
}
```

Both of these types are same and performance wise also they are equivalent. But they differ in their hoisting. While functions declared can be called from any part of the document, the functions written through expressions can only be called after they are defined. Otherwise the function expression becomes undefined.

11. What are JavaScript data types ?

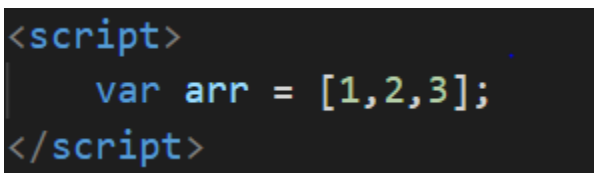
Ans:- JavaScript primarily has two types of data types.

- Primitive data types
- Non-primitive data types

12. What are the ways to create arrays in Javascript ?

Ans:- Following are the two conventional ways to create arrays in JavaScript.

1. By defining an array with elements.

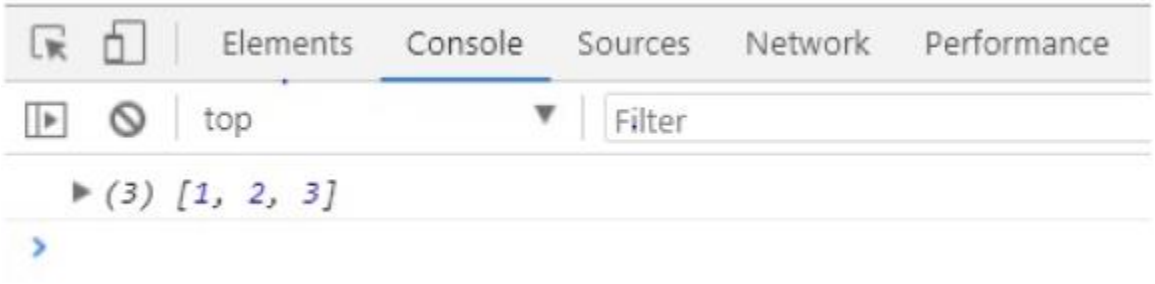


```
<script>
    var arr = [1,2,3];
</script>
```

2. By using the **new** operator.

```
<script>
    var arr = new Array(1,2,3);
    console.log(arr);
</script>
```

This will create an array with three elements like below.



13. How to remove all elements of an array ?

Ans:- There are several approaches to remove all elements from an array.

1. By re -assigning a blank array to the existing array.

```
<script>
    var arr = [1,2,3,4];
    arr = [];
    console.log(arr);
</script>
```

2. By equating the length of the array to zero

```
<script>
    var arr = [1,2,3,4];
    arr.length = 0;
    console.log(arr);
</script>
```

3. By using a conditional loop

```
<script>
    var arr = [1, 2, 3, 4];
    while (arr.length > 0) {
        arr.pop();
    }
    console.log(arr);
</script>
```

4. By using the splice() method with the length as array's length

```
<script>
    var arr = new Array(1,2,3);
    arr.splice(0,arr.length);
    console.log(arr);
</script>
```

14. What is the difference between undefined and null ?

Ans:- A variable becomes undefined when it is declared but yet not assigned a value.

```
<script>
    var myVar;
    console.log(myVar);
</script>
```

Or, when the variable is accessed before it is defined.

```
<script>
    console.log(myVar);
    var myVar;
</script>
```

A variable becomes null when it is assigned a **null** value explicitly.

15. What is the difference between == and === ?

Ans:- The main difference between == and === is that while == checks only for equality in value, === checks for both value and type.
This can be explained with the following example.

```
<script>
    console.log(2 == '2'); // true
    console.log(2 === '2'); // false
</script>
```

In the first console statement though the two values are of two different data types they are evaluated to be equal. But in the second console statement since both value and type are checked by the === operator and the expression evaluates to false.

16. What is the delete operator ?

Ans:- The delete operator or keyword deletes elements from array and variables. It has an inherent limitation that it can not delete variables declared with the var keyword.

17. What is void(0) in JavaScript ?

Ans:- void(0) is primarily used with anchor tags to keep the page from refreshing when a click handler is set on the same anchor tag.

```
<a href="javascript:void(0)" onclick="myFunc();">Click Me</a>
```

It is very helpful while dealing with cases where you want to execute the function without refreshing the page. here the developer does not has to really care whether the called function returns false or not.

18. How to detect the system details using JavaScript ?

Ans:- The navigator object which is a child of the window object contains all the details of the client system. Suppose for example we want to know the application name. Then we can get this value as follows.

```
<script>
    var os = navigator.appName;
    console.log(os);
</script>
```

19. What are the different scopes in JavaScript ?

Ans:- Basically there are three kinds of scopes in JavaScript.

1. Global scope:

The variable is said to be existing in the global scope if it is a direct child of the window object and can be accessed anywhere in the document. Following are ways to declare a global variable.

```
<script>
   var global1 = ''; // variable outside of enclosure
   window.global2 = ''; // variable attached to the window object
   function myFunc () {
       global3 = ''; // variable without the var keyword, global
   }                 // even though declared inside a function
```

From the above code snippet we can see that we can declare global variables in three different ways.

2. Local Scope:

The variables which are not direct child of the window object and can not be accessed outside of the block enclosing them are known to be having local scope. The following way we can declare variable in the local scope.

```
<script>
    function myFunc () {
        var myVar = 10;
    }
    // Outside of the function body we can not access the variable
</script>
```

3. Block level Scope:

Block level scope was introduced in ECMAScript6 version. Earlier local scope was only permitted for function blocks. Now we can have a new block level scope with in code blocks such as for loop, if conditionals etc. The block level scope is achieved using the **let** keyword.
The following example demonstrates how to achieve block level scope.
***It is a ES6 feature. So in order to test this you need to have a local setup in your machine that transpiles ES6 to ES5.

```
<script>
    function iterateArray() {
        var arr = [1, 2, 3, 4];
        for (let i = 0; i < arr.length; i++) {
            console.log(arr[i]);
        }
        // i is not accessible here as it has a block level scope
    }
</script>
```

20. What is a closure in JavaScript ?
A closure is an environment where functions are nested inside one another. The speciality of closure is that the inner function has access to the variables of the outer function.
Example:-

```
<script>
  function multiply (a){
      return function (b) {
          return a * b;
      }
  }
  console.log(multiply(5)(3)); // 15
</script>
```

Here as we can see the inner function has access to the variable **a** is a property of the parent function.

21. Write a javascript function that will produce the following output when invoked ?
console.log(mul (2)(3)(4)); // 24

Ans:- Below is the method which will produce the required output. It uses closure to achieve this functionality.

```
<script>
    function mul(a) {
        return function (b) {
            return function (c) {
                return a * b * c;
            }
        }
    }
    console.log(mul(2)(3)(4)); // 24
</script>
```

22. What is the output for the following code block ?

```
var output = (function(x){
delete x;
return x;
})(0);
console.log(output);
```

Ans:- The output is 0. This is an IIFE and x = 0; and delete method is only applicable to objects , it won't affect the value of x. So the output will be 0.

23. What is event propagation model in JavaScript ?

Ans:- By event propagation model in JavaScript we mean that when ever an event is triggered on any DOM node , it travels from child to parent or parent to child. There are basically two models.
1.Event Bubbling (Child element to parent element)
2.Event Capturing/ trickling(Parent element to child element)

*******Event Bubbling is the default behavior in JavaScript.

24. What is event bubbling ?

Ans:- Whenever there is an event triggered in the child element it propagates to the parent element. It is known as event bubbling. To understand this concept let us have different event handlers for same event e.g click event.

```
<body>
    <UL onclick="fromUL()">
        <LI onclick="fromLI()">Click Me!</LI>
        <LI onclick="fromLI()">Click Me!</LI>
        <LI onclick="fromLI()">Click Me!</LI>
    </UL>
    <script>
       function fromLI () {
           alert('I am from the LI element');
       }
       function fromUL () {
           alert('I am from the UL element');
       }
    </script>
</body>
```

Suppose we have UL and LI element that are parent and child. Now if we click on any LI element then first the alert will come from **fromLI()** function and and then the alert will come from **fromUL()** function. This demonstrates the fact that the click event is first triggered on the LI element due to user action and then it will be propagated to the UL element due to event bubbling phenomena.

25. What is event capturing ?

Ans:- Event capturing means whenever an event is triggered in the child element the event handler of the parent will execute first and then the event handler for the child will be triggered. In this mode we have to pass a third argument to the event listener as true.
The code snippet is as follows.

```
<body>
    <div id="div1">
        div 1
        <div id="div2" style="background-color: green;">
            div 2
        </div>
    </div>
    <script>
        var div1 = document.querySelector("#div1");
        var div2 = document.querySelector("#div2");

        div1.addEventListener("click", function (event) {
            alert("you clicked on div 1");
        }, true);

        div2.addEventListener("click", function (event) {
            alert("you clicked on div 2");
        }, false);
    </script>
</body>
```

As we can see the **addEventListener()** function is passed a third argument as true for the parent div element.

26. How many ways we can create object in Javascript ?

Ans:- There are three ways by which we can create objects in javascript.

1. By using object **literals**:

```
var obj = {}; // By using Object literals
```

or like this;

```
var obj = {
    name : 'Satya',
    profession: 'Software Engineer'
};
```

2. By using the **new** keyword:

```
var obj = new Object(); // By using new keyword
```

The **new** keyword is very important here as it will create an instance of the target object. It can be used with Array(), String() or Date() methods as well.
Another way, we can also pass some values to it as well like below example.

```
var obj = new Object({name: 'satya'}); // By using new keyword
```

3. By using **Object.create()** method:

```
<script>
    var obj = Object.create(Object); // By using  Object.create method
</script>
```

Above are the methods to create objects in JavaScript.

27. How to check if an object has a particular property in Javascript ?

Ans:- To ascertain that an object has a particular property there is an in built method named **hasOwnProperty()**. The below example demonstrates this.

```
<script>
    var obj = {
        name : 'Satyapriya Mishra',
        profession: 'Software Developer',
    };
    console.log(obj.hasOwnProperty('name')); // true
    console.log(obj.hasOwnProperty('age')); // false
</script>
```

28. How to get the constructor function of an object ?

Ans:- We can get the constructor name(the function from which an object is created/inherited) by the following way.

```
    <script>
        var obj = {
            name : 'Satyapriya Mishra',
            profession: 'Software Developer',
        };
        console.log(obj.constructor.name); // Object

    </script>
```

29. What is prototype in JavaScript ?

Ans:- Prototype is a very important concept is javascript. Mainly it serves two purposes. First it is used to add new properties and methods to an existing object and second, it is the model through which inheritance in javascript occurs.Let's start the discussion about prototype from a very basic example.

```
var arr = new Array(1,2,3,4);
arr.push(5);
console.log(arr); // [1,2,3,4,5]
```

The above example is a typical case of prototype inheritance. Let's try to understand it. Here **arr** is an array of type object which typically inherits the **Array()** constructor. In the second statement it tries to **push()** a new element into the array. But wait. Did we define the

push() method for this new arr. No. But how it is available to it.It is available because of the prototypal inheritance in javascript and the arr is inheriting the properties and methods from the Array() object.
Now let us dive into some real code and try to understand how prototype helps us in adding new properties to an existing object and how it helps us for inheritance.

```
<script>
    function myConstructor () {
        this.property1 = 1;
        this.property2 = 2;
    }
    var obj = new myConstructor();
    console.log(obj);
    myConstructor.prototype.property3 = 3; // Adding new property to the constructor
    console.log(obj.property3); // Property inheritance
</script>
```

Here **myConstructor()** is the constructor function which has two properties and is inherited by an object **obj**. Suppose we want to add a new property called property3 into the constructor. We can achieve this using prototypes by doing

myConstructor.prototype.property3 = 3;

Now we can verify that property3 is inherited to the obj by accessing the property from obj. This is how prototypes work in their very basic formation.

1. What is jQuery ?

Ans:- jQuery is a JavaScript Library created by John Resig. It simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. By using jQuery the syntax becomes very easier and developers can achieve the same level of functionality with very little code.

2. What are the main features of jQuery ?

Ans :- The main features of jQuery are listed below.

- **DOM manipulation** − The jQuery made it easy to select DOM elements, traverse them and modifying their content by using cross-browser open source selector engine called Sizzle.
- **Event handling** − Event handling in jQuery is very easy due to the syntax.
- **AJAX Support** − jQuery supports asynchronous REST methods.
- **Animations** − jQuery comes with plenty of built-in animation effects which we can use in websites.
- **Lightweight** − jQuery is a lightweight library - about 19KB in size (Minified and gzipped).
- **Cross Browser Support** − jQuery has cross browser compatibility and it is a major reason of its acceptance by the developer community.

3. How to use the jQuery library ?

Ans:- JQuery library can be included in the html file in three different ways.

- By using Content Delivery Network(CDN): In this way we attach the CDN link of jQuery library to the <script> tag. The advantage of using this method is that we don't have to ship the file of the library when we are exporting the application. It can be done int he following manner.

```
<body>
    <script src="https://code.jquery.com/jquery-2.2.4.min.js" integrity="sha256-BbhdlvQf/xTY9gja0Dq3HiwQF8LaCRTXxZKRutelT44="
        crossorigin="anonymous"></script>
</body>
```

- By downloading a local copy of the library: In this way we download the file of the library from the jQuery official website and place the file inside our application. Then we reference the file from the html file using <script> tags. This method is very useful for development purpose as it does not need internet connectivity while development.

- From npm : With modern day web applications getting complex , thanks to technologies like nodeJS , it is now super simple to add a library to the application through node package manager. For that we need to simply install the jQuery package and then import the library wherever we need to use the jQuery functions.

4. What is $?

Ans:- $ is the alias for jQuery and it is used as a handle while using the jQuery library. practically speaking $ and jQuery can be used interchangeably as handle and there is absolutely no difference.

5. What is document.ready() function ?

Ans:- As we know the primary role of jQuery is to manipulate the DOM. So by using document.ready() function we make sure that the document is ready before any of the jQuery methods are triggered by the event handlers, without which the browser may get unexpected results. When we use this document.ready() function , whenever the document is ready an event is triggered and subsequently the code inside the ready function executes. It is always advisable to use the document.ready() function while writing jQuery code.
An example of document.ready() is given below.

```
<script>
    $(document).ready(function () {
        // jQuery code goes here
    })
</script>
```

6. What is the difference between document.ready() and window.onload() ?

Ans:- The main difference between document.ready() and window.onload() are as follows.
ready() function is fired when the DOM (Document Object Model) is loaded. This happens when the entire markup is parsed by the browser and the text content are rendered except the images and the links while the onload() event is fired when all the content are rendered including the images and the links. So obviously the code inside the window.onload() is executed after the code inside the document.ready() is executed.

7. What are the different type of selectors in Jquery?

Ans:- We can select DOM nodes in various ways in jQuery. Following are some of the selectors we can use to select elements.

- **By tag name:** We can select the elements using their tag names like in the following manner.

```
<body>
    <p>I am a paragraph!</p>
    <button> Click Me!</button>
    <script src="https://code.jquery.com/jquery-2.2.4.min.js" integrity="sha256-Bbhdlvqf/xTY9gja0
        crossorigin="anonymous"></script>

        <script>
            $(document).ready(function () {
                $('button').click(function () {
                    $('p').text('I am a tag selected element');
                });
            })
        </script>
</body>
```

As we can see we select the button element and the paragraph element by their respective tag names. When we click the button the text inside the paragraph element changes.

- **By class name:** We can select elements based on their class names in the following manner. If we select the element by class name, jQuery returns us the entire list of elements having the concerned class name.

```
<body>
    <p class="paragraph">I am a paragraph!</p>
    <p class="paragraph">I am also a paragraph with the same class.</p>
        <script>
            $(document).ready(function () {
                var paragraphsArray = $('.paragraph');
                console.log(paragraphsArray);
            })
        </script>
</body>
```

If you pay attention you can see that we are using a **.**(dot) for selecting the element which is highlighted in the above image.

We can select multiple elements with different class names by providing the comma separated list of classes as the first argument to the jQuery handle which is demonstrated in the following example.

```
<body>
    <p class="para1">I am a paragraph!</p>
    <p class="para2">I am also a paragraph with the same class.</p>
        <script>
            $(document).ready(function () {
                var paragraphsArray = $('.para1, .para2');
                console.log(paragraphsArray);
            })
        </script>
</body>
```

- **By id values:** We can select DOM nodes from id selectors. An example in this respect is demonstrated below.

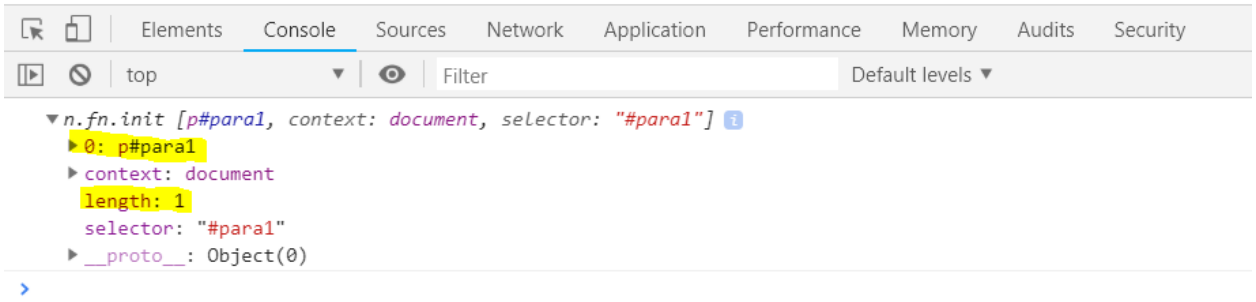
```
<body>
    <p id="para1">I am a paragraph!</p>
        <script>
            $(document).ready(function () {
                var paragraph = $('#para1');
                console.log(paragraph);
            })
        </script>
</body>
```

From the highlighted part we can see that for selecting elements by referencing their id values , we need to use the # (hash) symbol.

Now let's understand a very tricky situation. Like in the previous example , suppose we have two DOM elements with the same id (for demo purpose only, ideally we should not do it). Now when we try to select the elements with the id selector **,** jQuery will return us the first matching element. This is demonstrated by the following example.

```
<body>
    <p id="para1">I am a paragraph!</p>
    <p id="para1">I am a paragraph!</p>
        <script>
            $(document).ready(function () {
                var paragraph = $('#para1');
                console.log(paragraph);
            })
        </script>
</body>
```

When we run the above code snippet we will have the following output in the console window of the browser.



Here the length attribute of the array is 1. So we can safely say that only one element is selected even though we have multiple elements with the same id value.

- **By attributes:** We can select DOM nodes with attribute selector also. In this way of selection we can choose from practically any type of attribute. This method of selecting returns an array of all the matching elements irrespective of their attribute types.

```
<body>
    <p data="para1">I am a paragraph!</p>
    <p data="para1">I am a paragraph!</p>
    <script>
        $(document).ready(function () {
            var paragraph = $("[data=para1]");
            console.log(paragraph);
        })
    </script>
</body>
```

Please observe the highlighted syntax for selecting element with the attribute selector. The typical syntax is **$("attr = value")**.

8. What is the difference between width() and css('width') in jQuery ?

Ans:- The main difference between width() and css('width') is the data type of the value they return or set. While using css('width') we need to explicitly use the unit of width such as px , but for width() we just pass a number without having to specify any unit. Examples of each of the methods are given below.

```
<body>
    <div></div>
    <button>Click Me!</button>
    <script>
        $(document).ready(function () {
            $("button").click(function () {
                $('div').width(20);
            });
        });
    </script>
</body>
```

```
<body>
    <div></div>
    <button>Click Me!</button>
    <script>
        $(document).ready(function () {
            $("button").click(function () {
                $('div').css('width', '20px');
            });
        });
    </script>
</body>
```

We can mark the difference in the above two code snippets as in the css('width') method we are explicitly providing the unit as px while for the width() method we dont need to specify the unit.

9. What is the significance of $.on() method ?

Ans:- The $.on() method is used for two purposes.

- First it is used to bind events on elements like in the following example.

```
<body>
    <div></div>
    <button>Click Me!</button>
    <script>
        $(document).ready(function () {
            $("button").on('click',function () {
                $('div').css('width', '20px');
            });
        });
    </script>
</body>
```

Here we can see that we are binding the click event of the button with the on() method.

- Second it is used to set event listeners for DOM nodes which are appended to the DOM later than the first load of the DOM. For example suppose we add one new element to the DOM due to user interaction. Now if we want to manipulate the newly added element by a button click, then jQuery.click() method will not be able to detect this element as it is added later. So in this scenario we have to use the $.on() method to track that newly added element and manipulate it.

10. What is $.param() method ?

Ans:- $.param() method is used to serialize an array or an object. This method is particularly useful while sending data to the server through http requests. The implementation of the $.param() method is demonstrated below.

```
<body>
    <div></div>
    <button>Click Me!</button>
    <script>
        $(document).ready(function () {
            personObj = new Object();
            personObj.firstname = "Satyapriya";
            personObj.lastname = "Mishra";
            personObj.age = 26;
            personObj.eyecolor = "black";
            $("button").click(function () {
                $("div").text($.param(personObj));
            });
        });
    </script>
</body>
```

Here we serialize an object and use the serialized object as text to the div element. The output on the browser is as follows.

firstname=Satyapriya&lastname=Mishra&age=26&eyecolor=black

11. What is the difference between $(this) and this ?

Ans:- There is not much difference between the two but the context in which they are getting used. When we use $(this), the current element becomes an jQuery element automatically and we can use different jQuery methods on it. But with 'this', the element becomes an JavaScript object and we can't use jQuery methods on that object. An example is given below.

```
<body>
    <div></div>
    <button>Click Me!</button>
    <script>
        $(document).ready(function () {
            $('button').click(function () {
                alert($(this).text());
                alert(this.innerText); // works as expected
                alert(this.text()); // throws an error
            });
        });
    </script>
</body>
```

12. What is the difference between size an length ?

Ans:- In jQuery both size and length do the same thing yet a bit differently. size is a method which returns the number of number of elements in an array of objects , while length is a property which returns the number of elements in the array. Since length is a property it works faster.4

13. Can we use multiple document.ready() in jQuery ?

Ans:- Yes, we can use multiple ready functions.

14. What Is "noconflict" Method In Jquery?

Ans:- Jquery.Conflict method is used in case other client side libraries use the **$** symbol as their handle along with Jquery then we will use this method. It prevents collision of handles in between the two libraries and we can specify another handle for jQuery so that it does not collide with the other library.

15. How We Can Check if Element is Empty In Jquery?

Ans:- We can check if element is empty by the is() method of jquery. The implementation of the method is given below.

```
<script>
    $(document).ready(function () {
        if ($('element').is(':empty')) {
            // code for the next step
        }
    }
    )
</script>
```

16. How We Can Check Element Exists Or Not In Jquery ?

Ans:- The most convenient way to check if an element exists is to check the length of the element. If it does have a length greater than 0, then it exists, otherwise not. The code snippet below demonstrates this.

```
<script>
    $(document).ready(function () {
        if ($('element').length > 0 ){
            // element exists
        } else {
            // element does not exist
        }
    }
    )
</script>
```

17. What are the differences between “parents” And “parent” methods?

Ans:- “parents” method is used to traverse all along the DOM tree, whereas “parent” method is used to traverse only one level.

18. What is the empty() method in jQuery ?

Ans:- The empty() method removes all child nodes and content from the selected elements.
It removes the content and the child nodes of the selected element(s) , but not the element itself. Below code snippet demonstrates this method.

```
<script>

    $(document).ready(function () {
        $('.btn').click(function () {
            $('div').empty();
        });
    })

</script>
```

19. How we can check/uncheck checkbox in Jquery?

Ans:- We can check and uncheck checkboxes by attributes. The code below shows this.

```
<script>

    $(document).ready(function () {
        $('#checkBox').attr('checked', true);
        $('#checkBox').attr('checked', false);
    })

</script>
```

20. What is .clone() method in jQuery ?

Ans:- .clone() method is used to clone/copy matched elements. We need to append/prepend the cloned elements to the DOM to make them a part of the DOM. Below example shows the use of .clone() method.

```
<script>

    $(document).ready(function () {
        $('#btn').click(function () {
            $('#ele').clone().prependTo('div');
        });
    })

</script>
```

21. What are the methods to provide basic animations in jQuery ?

Ans:- The most common methods which are used to give animation effects to jQuery elements are listed below.

- toggle()
- fadeIn()
- fadeOut()
- slideUp()
- slideDown()
- hide()
- show()

22. Give an example to animate an element in jQuery ?

Ans:- .animate() method is used to give animation effects to jQuery elements.

```
<script>

    $(document).ready(function () {
        $("#element").animate({width:"300px"});
    })

</script>
```

23. What is chaining in jQuery ?

Ans:- By chaining we can chain multiple methods one after another so that they get executed sequentially. It is a very efficient way of writing jQuery code.
An example of method chaining is given below.

```
<script>
    $(document).ready(function () {
        $("#element").hide().show().addClass('newClass').css('color', 'red');
    })
</script>
```

24. How to make an basic ajax request to fetch data from an external source ?

Ans:- AJAX stands for Asynchronous JavaScript and XML. This method is used to make **http** requests to the server. Below code snippet gives an example of the most bare bone format of an AJAX request made to fetch some data from an external website.

```
<script>
    $(document).ready(function () {
        $.ajax({
            url: "https://jsonplaceholder.typicode.com/posts",
            success: function (result) {
                console.log(result);
            }
        });
    })
</script>
```

25. How to get the value of the selected element from a dropdown using jQuery ?

Ans:- We can get the value of the selected dropdown field by setting event handler for the **onChange** event and then using the attribute **selected**. The complete code snippet for the operation is given below. Mark the highlighted parts.

```
<select id="dropDownId">
    <option>1</option>
    <option>2</option>
</select>
<script>
    $(document).ready(function () {
        $('#dropDownId').change(function () {
            var val = $('#dropDownId :selected').text();
            alert(val);
        });
    })
</script>
```

1. What is AngularJS ?

Ans:- AngularJS is a super heroic javaScript framework built for building highly scalable single page applications. AngularJS provides us with those features which HTML should have given, had it been designed for creating dynamic web pages. AngularJS is built upon the MVW/ MV* architechture where M, V ad W/* represent Model, View and Whatever.
We can also build angular applications in different design patterns like MVVM, MVP, MVC etc as per the requirements.

2. What are the most important features of AngularJS ?

Ans:- The most important features of AngularJS are listed below.

- One way/ two way data binding
- Services
- Routing
- Directives
- RESTful API
- Controllers
- Scope
- Model
- View
- Filters

3. How to use AngularJS in our application ?

Ans:- We can use angularjs in our application in two different ways.

- by downloading a local copy of the angularjs framework file and then referencing it from the html file where we want to use it.
- by accessing the angularjs file through the Content Delivery Network (CDN).

4. What is data binding in angularjs ?

Ans:- Data binding is the synchronization of data between the model and the view. Here since mostly the data resides inside the controllers and due to the design pattern constraints controllers and view are separated from each other. So data binding in angularjs is achieved by **scope** object. The data binding in angularjs is designed in such a way that the model is the single source of truth and the view is just a reflection of the data contained in the model.
There are two ways of data binding in angularjs.

- One way data binding
- Two way data binding

In one way data binding data from model is reflected on the view, but there is no possibility of changing the data from the view to the model.
In two way data binding data in the model as well as the view are in sync i.e we can change the data from the view and it will change the data in the model.

5. What are directives in angularjs ?

Ans:- Directives are a very important feature in angularjs and one of the most important reasons for angularjs being so popular. By directives we can create dynamic html tags with the capability of event listening. There are basically two types of directives in terms of creation.

- In built directives (ng-app, ng-bind, ng-controller etc)
- Custom directives

Any directive in angular can take any one of the following four forms.

- Element directive
- Attribute directive
- Class directive
- Comment directive

6. What is the template in angularjs ?

Ans:- Template represents the view part of the angular application. It contains the html markup, the bindings and the custom directives. For template we can directly specify the template inline or we can refer a specific file with the templateUrl property from the router.

7. What is $scope ?

Ans:- In MVC design patter the controller cannot have direct access to the view or vice versa. But the view needs to access the properties and methods of the controller in terms of data and event binding. To address this problem in angularjs we have $scope. Here $scope acts as a glue between the controller and the view. For each controller automatically a scope object is created which points to the methods and properties of that particular controller. Whenever an angular application is created a top most scope namely the $rootScope is created at the global level.

8. What are directives ?

Ans:- A directive is a special markup which is used to add some special behavior to html tags/ elements. In other words with directives we can add dynamic behavior to html tags which are otherwise static and also we can create our own custom tags.
In angularjs directives can be used in four types.

- E-Element directives
- C-Class directives
- M-Comment directives
- A-Attribute directives

9. What is a service ?

Ans:- A service is a function or class which does some specific task. The service is dependency injected into the controller or directive to be used.
e.g - $http service, $loaction service etc

10. What are the advantages of service ?

Ans:- Following are some of the advantages of services.

- Code Decoupling - By services we can decouple the code of the controllers from the services and we can easily separate them or change code without having to change a lot of code
- Code Maintainability - Thanks to services the code base becomes more maintainable for large scale applications
- Code testability - Code is more testable and we can perform unit tests on both the controllers as well as the services separately

- Code Scalability **-** The application becomes more scalable since the code is organized.

11. What is ng-app ?

Ans:- ng-app is a directive which denotes the starting point of an angularjs application. Whenever the browser encounters an ng-app directive, the elements inside that are considered to be part of angularjs. Any element outside of the ng-app element will have no binding to angularjs.

```
<body ng-app = ''>

</body>
```

12. What is ng-model ?

Ans:- ng-model directive is used to bind data in the view. An example is given below.

```
<script>
    var app = angular.module('app', []);
    app.controller('ctrl', function ($scope) {
        $scope.name = 'Satyapriya Mishra';
    });
</script>
```

Suppose we have an application with the name **app** and a controller attached to it namely **ctrl**. The controller has a property called **name** attached to the $scope object. Now if we want to have a two way data binding with the view to the controller and vice versa then we would have a view like the below code snippet.

```
<body ng-app = "app">
    <div ng-controller = "ctrl">
        <input type="text" ng-model="name">
        {{name}}
    </div>
```

Here inside the input box if we change the value, then due to ng-model binding this new value will be reflected inside the controller. On the other hand due to expression binding we will get the new value bound to the view .

13. What is expression in angularjs ?

Ans:- Expression is used to bind data in the view segment of the application. It is used for one way binding. Anything inside the double curly braces of the expressions will be evaluated to one singe value. So we can also have different kinds of expressions in angularjs. Some examples are given below.

```
<body ng-app = "app">
    <div ng-controller = "ctrl">
        {{name}}
    </div>
   <script>
        var app = angular.module('app', []);
        app.controller('ctrl', function ($scope) {
            $scope.name = 'Satyapriya Mishra';
        });
   </script>
</body>
```

If we look at the highlighted part we can see that the name property from the controller is bound to the view by means of expression.

The below example demonstrates a very generic use of expressions to evaluate any given expression.

```
<div>

    {{2 + 2}}

</div>
```

The below example demonstrates the use of expressions with functions. Here we can invoke functions with a return value to bind that returned value to the view segment.

```
<body ng-app = "app">
    <div ng-controller = "ctrl">
        {{getName()}}
    </div>
   <script>
        var app = angular.module('app', []);
        app.controller('ctrl', function ($scope) {
            $scope.name = 'Satyapriya Mishra';
            $scope.getName = function () {
                return $scope.name;
            };
        });
   </script>
</body>
```

14. What is the difference between ng-bind and ng-model ?

Ans:- There is a significant difference ng-bind and ng-model. ng-bind is used for one way data binding i.e $scope => view , but ng-model facilitates two way data binding i.e $scope => view and view => $scope.

15. What are the ways to share data between two controllers ?

Ans:- There are primary four ways to share data between two controllwers which are listed below.

- Events
- $parent, controllerAs
- $rootScope
- Services

Out of these four ways of data sharing between controllers, the data sharing using services is ost preferred due to the fact that the by using services the code becomes decoupled and it is more testable.

16. What are the different tools to test an angularjs application ?

Ans:- There are various tools to test angularjs applications. Listed below are some of the most popular tools.

- Karma
- Jasmine
- protractor

17. What is angularjs digest cycle ?

Ans:- Digest cycle is the process wherein angularjs keeps a tab on each of the bound values in the model and whenever they change it runs a cycle to replace the old values with the new values. The digest cycle is comprised of three steps listed below.

- $watch
- $digest
- $apply

The typical syntax of a digest cycle is given below.

```
$scope.$watch('aModel', function (newValue, oldValue) {
    //update the DOM with newValue
});
```

If we want to trigger a digest cycle manually, then we have to use $apply. The typical use case of using $apply manually is when we want to change some model value inside a setTimeout() function where the context changes. In this case we need to manually trigger the $apply() method to make the changed model data be in sync with the rest of the application.

18. What is internationalization in angularjs ?

Ans:- Internationalization of angularjs means to show locale specific data in an application.
e.g language

19. What is the difference between angularjs and javascript expressions ?

Ans:- The difference between angularjs and javascript expression are listed below.

- Angular expressions can have html in them, where javascript expressions do not
- Angular expressions support filters, but javascript expressions do not

20. What is an angular module ?

Ans:- In general an angular module is a self contained application. But for large scale applications there are multiple modules which are part of a parent module to work independently.

An angular module is registered through the ng-app directive and it needs to be initialized in the script as follows.

```
<body ng-app="app">
    <div ng-controller="ctrl">
    </div>
    <script>
        var app = angular.module('app', []);
    </script>
</body>
```

This is an angular module in the most basic bare bone format. But if we want to attach controller, service and directive to this app module , we can do like below.

```
<body ng-app="app">
    <div ng-controller="ctrl">
    </div>
    <script>
        var app = angular.module('app', []);
        app.controller ('ctrl',function ($scope) {
            // code for controller goes here
        });

        app.directive('myDirective', function () {
            // code for directive goes here
        });

        app.service ('myService', function () {
            // code for service goes here
        });
    </script>
</body>
```

Here we can see that for attaching a controller, directive and service we attach them to the module object with the dot(**.**) operator and then the first parameter is the name of the component and the second parameter is the callback function containing the logic of that specific component.

21. What are the scopes available for a directive ?

Ans:- There are primarily two scopes attached to the directives in angularjs.

- Shared scope
- isolated scope

Shared scope is that scope when a directive shares its properties and methods with the rest of the application. In isolated scope the properties and methods become private to that concerned directive only and they don't get shared with the rest of the application. Shared scoping is the default behavior of the directive scope.

22. How to isolate a directives scope in angularjs ?

Ans:- Isolating a directives scope in angularjs is pretty straight-forward. We need to pass an object to the scope property of the directive definition object. The simplest way to do so is by passing a null object to the DDO. An example for this is given below.

```
app.directive('myDirective', function () {
    return {
        restrict: "A",
        template: "<h1>Made by a directive!</h1>",
        scope: {}
    };
});
```

23. How to force an angularjs service to return a promise ?

Ans:- We can force the angularjs service to return a promise by the $q service. By using $q we can have states like defer and resolve. A very simple code snippet for the above operation is given below.

```
angular.factory('testService', function ($q) {
    return {
        getName: function () {
            var deferred = $q.defer();

            myAPI.getData().then(function (data) {
                deferred.resolve(data)
            })

            return deferred.promise;
        }
    }
})
```

24. Give an example of angularjs routing ?

Ans:- Routing is a very important feature in angularjs. By virtue of routing only we can create Single Page Applications(SPA) in angularjs.
The below example demonstrates the different parts of an angular routing operation.

- First we include the angular router file in the application. It has be included after the angularjs file is included and it should be of the same version as the angularjs version.

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular-route.js"></script>
```

- Second we create the main as well as he child routes. These are hyper links which the user will click to move to different routes.

```
<a href="#/!">Main</a>

<a href="#!one">One</a>
<a href="#!two">Two</a>
<a href="#!three">Three</a>
```

Third step we inject the ngRoute into the application module so that we can use its properties and methods in our application.

```
var app = angular.module("app", ["ngRoute"]);
```

Fourth step we invoke the angular config() method and inject the $routeProvider to configure our routes.

```
app.config(function ($routeProvider) {
    $routeProvider
        .when("/", {
            templateUrl: "main.html"
        })
        .when("/one", {
            templateUrl: "one.html"
        })
        .when("/two", {
            templateUrl: "two.html"
        })
        .when("/three", {
            templateUrl: "three.html"
        });
```

Fifth step we configure the different routes. We can provide the template Urls and the controllers for each of the specified routes.

25. What is dependency injection ?

Ans:- Dependency Injection is a software design architecture wherein an independent code block is injected into an dependent code block by reference without having to write the entire code in the same block as the dependent code block.
By dependency injection we achieve code decoupling and abstraction. The module where the other module is injected is not aware of the internal architecture of the independent module.
The best example for dependency injection in angularjs is Services. In the following example an in built $http service is used to demonstrate this concept.

```
<script>

    var app = angular.module('myApp', []);
    app.controller('myCtrl', function ($scope, $http) {
        $http.get("myAPIUrl").then(function (response) {
            $scope.data = response.data;
        });
    });
</script>
```

Here as we can see the $http service is dependency injected into the controller and the controller is not aware of the internal architecture of the $http service. Any ,time we want to separate the service

from the controller we can do so by simply removing the dependency injection. Due to these facts dependency injection is so popular in software design architectures.

26. What is the difference between $rootScope and $scope ?

Ans:- Both $rootScope and $scope are scope objects but still there are some differences between them which are highlighted below.

There is only one $rootScope in the application but we can have unlimited number of $scope in the application. $rootScope is the parent of all the $scope objects.

Whenever an application module is created a $rootScope is created but whenever a controller is created a $scope is created.

The data models inside the $rootScope is available globally to all controllers and directives but the data inside the $scope object is available only to the current controller or its child controllers by means of inheritance.

27. How to enable html5 mode in angularjs ?

Ans:- The html5 mode is very useful while prettifying ugly urls and to avoid special charecters like # in the route urls. The following example illustrates the code to enable html5 mode in angularjs.

```
<script>
    angular.module('myApp', [])

        .config(function ($routeProvider, $locationProvider) {

            $routeProvider
                .when('/', {
                    templateUrl: 'partials/home.html',
                    controller: mainController
                })
            $locationProvider.html5Mode(true);
        });
</script>
```

We need to pass **true** value to the html5Mode method to enable html5 mode in an angularjs application.

28. What is the difference between $ and $$?

Ans:- In angularjs code by convention, to denote public and private objects we refix them with $ and $$ respectively.

29. How to access the properties of parent controller form the child controller scope ?

Ans:- We can access the properties of the parent controller with the use of $parent. For data binding and accessing we can directly use $parent form the html template like the following example.

```
<body ng-app="app">

    <div ng-controller="ParentCtrl">
        <h1>{{ name }}</h1>
        <p>{{ age }}</p>
        <div ng-controller="ChildCtrl">
            <h1>{{ title }}</h1>
            <input type="text" ng-model="$parent.name" />
        </div>

        <script>
            var app = angular.module('app', []);
            app.controller('ParentCtrl', function ($scope) {
                $scope.name = 'Satya';
                $scope.age = 27;
            });
            app.controller('ChildCtrl', function ($scope) {
                $scope.title = 'Scope Inheritance';
            });
        </script>
</body>
```

If we look at the highlighted part we can see that inside the scope of the child controller **ChildCtrl** in the view we are able to access the name property of the parent controller **ParentCtrl** with the help of $parent. The $parent object contains all the properties and methods of the arent controller scope.

30. What is the difference between compile and link functions in Directive Definition Object in angularjs ?

Ans:- The main difference between a compile and link function is that the compile function does the directive template manipulation after it is cloned whereas the link function acts as the link between the template mand the directive definition object to listen for the events triggered by the user, by setting the event listeners. Link function is executed after compile function clones the templates in to the Document Object Model.

31. What is the difference between ng-if and ng-show ?

Ans:- There is a very significant difference between the ng-if and ng-show directives.
If the condition is false ng-show simply make it invisible by adding a css display: none property to the element. But when the condition is false ng-if completely removes the element from the DOM. The template rendered by the use of ng-if loads faster than the template rendered by use of ng-show.

32. Give an example of ng-if ?

Ans:- An example for ng-if directive is given below.

```
<body ng-app="app">

    <div ng-controller="Ctrl">
        <div ng-if="status">
            <h2>Heading inside the directive</h2>
        </div>
    </div>


    <script>
        var app = angular.module('app', []);
        app.controller('Ctrl', function ($scope) {
            $scope.status = true;
        });
    </script>
</body>
```

Here in the highlighted part we can see that ng-if directive is used and we are assigning an expression form the controller. When the value is evaluated to true , the content inside the div element will be visible , else it will be removed entirely from the DOM.

33. Give an example of ng-show ?

Ans:- An example of ng-show is given below.

```
    <div ng-controller="Ctrl">
        <div ng-show="status">
            <h2>Heading inside the directive</h2>
        </div>
    </div>


    <script>
        var app = angular.module('app', []);
        app.controller('Ctrl', function ($scope) {
            $scope.status = false;
        });
    </script>
```

Similar to ng-if , ng-show hides the element from the DOM if the expression if evaluated to false. But it does not remove the element from the DOM, rather it attached a display none property to the

concerned element. When we look at the browser developer tool we can have an image like the below.

34. What is ng-repeat and give an example of it ?

Ans:- Given below is an example of ng-repeat which is used to iterate over a record array to display them on the DOM.

```
<div ng-controller="Ctrl">
    <ul ng-repeat = "x in records">
        <li>{{x}}</li>
    </ul>
</div>


<script>
    var app = angular.module('app', []);
    app.controller('Ctrl', function ($scope) {
        $scope.records = [
            "Alfreds Futterkiste",
            "Berglunds snabbköp",
            "Centro comercial Moctezuma",
            "Ernst Handel",
        ]
    });
</script>
```

35. Demonstrate a custom service with an example ?

Ans:- The following example demonstrates the example of a custom service. Here we create a custom service and then invoke the service from inside of the controller after dependency injection.

```
<div ng-controller="Ctrl">
    <button ng-click="alertMethod()">Click Me</button>
</div>


<script>
    var app = angular.module('app', []);
    app.controller('Ctrl', function ($scope, alertService) {
        $scope.alertMethod = function () {
            alertService.alertFunc('I am coming from a service');
        }
    });
    app.service('alertService', function () {
        this.alertFunc = function (msg) {
            alert(msg);
        }
    });
</script>
```

36. What is ng-transclude ?

Ans:- When we are creating a Directive Definition Object with a template inside it, then wherever the directive is used content of the original element will be lost and the template of the custom directive will replace those contents. This is a disadvantage as in that case we lose the ability to reuse the custom directive in multiple instances.
To get rid of this disadvantage angularjs provides us with a directive called ng-transclude. By using ng-transclude , it acts as a placeholder of the original contents of the element so that they don't get lost during the DOM formation phase.

37. Give an example of ng-transclude ?

Ans:- An example for ng-transclude is given below.

Let's look at the view segment first.

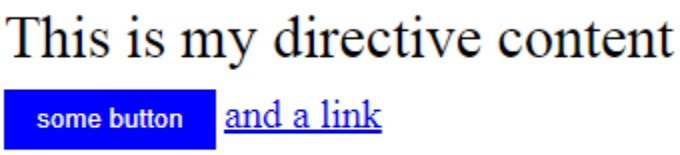
```
<div ng-controller="Ctrl">
    <div my-directive>
        <button>some button</button>
        <a href="#">and a link</a>
    </div>
</div>
```

Here we can see that the custom directive is used as a attribute directive here for the div element. The div element also contains some other child nodes like button and anchor tag.

Now let's look at the script part where we will find the implementation of the ng-transclude, which is highlighted.

```
<script>
    var app = angular.module('app', []);
    app.controller('Ctrl', function ($scope) {
    });
    app.directive('myDirective', function () {
        return {
            transclude: true,
            template: '<div class="something">Template</div> <ng-transclude></ng-transclude>'
        }
    });
</script>
```

When we use ng-transclude in the browser we can see that both the contents i.e from the custom directive as well as from the original div element in the view are rendered.



38. What is track by in angularjs ?

Ans:- Suppose we have a case where we want to iterate over a large collection of data using ng-repeat. If some of our data changes due to the server pushing some new sets of records , then due to angularjs life cycle hook, our entire DOM will be re-rendered and that will be a huge cost to the server. To avoid this we use the **track by** expression where in we track each record with some identifier such as index(must be unique) and if that record is not updated ,then ng-repeat will not re-render this part of the DOM. **track by** is very important when we are working with data intensive application. It boosts the performance. It also helps to prevent duplicate entries in the records.

39. What is the difference between config() and run() methods in angularjs ?

Ans:- Config– This block is executed during the provider registration and configuration phase. Only providers and constants can be injected into configuration blocks. This block is used to inject module wise configuration settings to prevent accidental instantiation of services before they have been fully configured. This block is created using config() method.

Run – This block is executed after the configuration block. It is used to inject instances and constants. This block is created using run() method. This method is like as main method in C or C++. The run block is a great place to put event handlers that need to be executed at the root level for the application. For example, authentication handlers.

40. How to start an angularjs application manually ?

Ans:- We can manually start an angularjs application by using the bootstrap method. The following example illustrates this concept.

```
<body>
    <div ng-controller="ctrl">
        {{name}}
    </div>
    <script>
        var app = angular.module('demo', [])
            .controller('ctrl', function ($scope) {
                $scope.name = 'Satyapriya!';
            });
        angular.bootstrap(document, ['demo']);
    </script>
</body>
```

41. How to have multiple modules in an angulrjs application ?

Ans:- Since an angular module essentially refers to a separate application, to run two separate modules simultaneously we need to bootstrap manually. The following example demonstrates this concept.

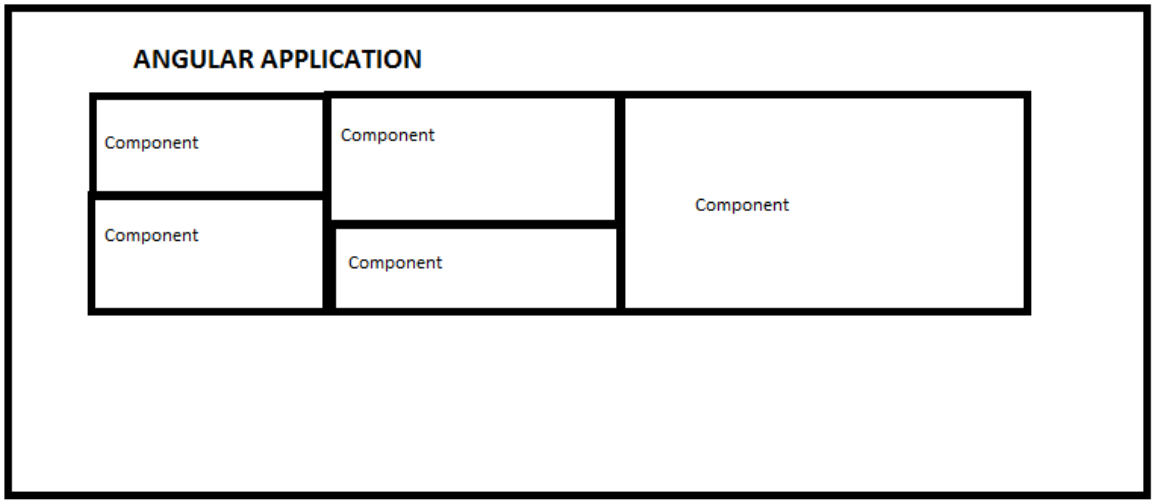
```
<body>
    <div id="App1" ng-app="app1" ng-controller="app1Ctrl">
        {{name}}
    </div>

    <div id="App2" ng-app="app2" ng-controller="app2Ctrl">
        {{address}}
    </div>
    <script>
        var app1 = angular.module("app1", [])
        app1.controller("app1Ctrl",
            function ($scope) {
                $scope.name = 'Satyapriya Mishra';
            }
        );
        var app2 = angular.module("app2", [])
        app2.controller("app2Ctrl",
            function ($scope) {
                $scope.address = 'Bengaluru';
            }
        );
        angular.bootstrap(document.getElementById("App2"), ['app2']);
    </script>
</body>
```

1. What is Angular ?

Ans:- Angular is the latest version of Angular framework that include version 2/4/5/6/7 etc. The creators of Angular have removed JS suffix from the framework as this version of the framework works best with Typescript which is a superset of JavaScript with a component driven architecture.

2. What is a component in angular ?

Ans:- A component is a part of an application that controls a patch of the view. In an angular application components combine to give rise to a complete application. By components we can attach properties, methods and event handlers to the view. Components consume different services to attach functionality to the angular application. Components are designated with the @Component decorator, which is in fact a decorator. A typical structure of an angular application is given below.



3. Ideally what are the files inside a component ?

Ans:- Whenever a new component is created it should have the following files. Let's suppose that our component name is header component.

- header.component.html
- header.component.css
- header.component.spec.ts
- header.component.ts

The header.component.html file contains the view part. Here we do all the bindings and set the events. This is the view that is rendered wherever the component is instantiated. The css file takes care of the styles of the markup. It contains only the styles of the concerned component. The spec file contains the unit tests for the component. The header.component.ts file contains the decorators, metadata and the business logic for this component. It is the file where external services are instantiated and consumed.

4. What are component decorators in angular ?

Ans:- Component decorator allows you to mark a class as an Angular component and provide additional metadata that determines how the component should be processed, instantiated and used at runtime.
The main function of decorator is to provide angular some metadata so that angular can decide what to do with the typescript class. Internally angular implements the decorator functionality by using the reflect-metadata node package.
There are four kinds of decorators.

- Class decorators
- Property decorators
- Method decorators
- Parameter decorators

An example of a class decorator is given below. Here we can see that we pass certain parameters like selector, templteUrl and styleUrls to the decorator function so that angular will be able to get these metadata and convert the class into a component.

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
```

5. What is the command to create a new component ?

Ans:- Using angular command line interface to create a new component is super easy. We just need the following command and angular will create a brand new component for us, all files included.

```
ng generate component newComponent
```

Here newComponent is the name of the component we want to create. There is also a short hand version of this command. The short hand command is given below.

```
ng g c  newComponent
```

Here g stands for generate and c stands for component.

6. What is ngFor and give an example of it ?

Ans:- In angular ngFor is a built in directive which is used for iterating over a collection of items. The items may be a simple array or an array of objects. The collection is written inside the component and the ngFor directive is used inside the view of the application. This directive creates an instance of each of the individual record and renders it in the view. An example is given below.

For the purpose of simplicity we have taken a simple array of objects as the records in the component.

```
export class AppComponent {
  records = [
    {
      name: 'Satya',
      age: 27
    },
    {
      name: 'Mitra',
      age: 28
    },
    {
      name: 'Saurabh',
      age: 27
    },
  ];
}
```

Now in the view part i.e in the html template we can use the ngFor directive as follows.

```
<ul *ngFor = "let item of records">
  <li>{{item.name}} is {{item.age}} years old.</li>
</ul>
```

Here we can see that when we are iterating over the record, we take into consideration each element of the record as item and do the bindings on one item at a time.

7. How to get the index of each item in ngFor ?

Ans:- Some times to keep track of the items we need to keep a track of the items in the view which can be done by getting the index of the item in the list or records. The following example demonstrates the code for getting the index of each item in the records.

```
<ul *ngFor = "let item of records; index as i ">
  <li>{{i + 1 }}.{{item.name}} is {{item.age}} years old.</li>
</ul>
```

8. What are the different ways we can do one way data binding in angular ?

Ans:- Basically there are three ways we can do one way data binding in angular. Explanation and code snippet for each of them are given below.

By interpolation: We can do one way data binding through interpolation. Here we used two nested curly braces to bind the expression from the component to the view. An example is given below.

```
<input type="submit" class="btn" value="{{texts}}">
```

By expression binding : In this way we bind one attribute of the element to a value from the component. Same example with expression binding is given below.

```
<input type="submit" class="btn" [value]="texts">
```

By bind keyword: We can do one way data binding through the bind keyword. Here we have to prefix the keyword with the attribute value to bind the data. An example is given below.

```
<input type="submit" class="btn" bind-value="texts">
```

9. How to do two way data binding in angular ?

Ans:- The process where the component and view data are in sync with each other when any of them changes, is known as two way data binding. Two way data binding in angular is done through the ngModel directive. An example of two way data binding is given below.

```
<input type="text" [(ngModel)] = "texts">
```

Internally there is no such thing as two way data binding in angular version >= 2. The above expression is actually a property binding and an event binding. The above code snippet can be broken up into two separate bindings as follows.

```
<input [ngModel]="texts" (ngModelChange)="texts = $event">
```

10. What is ngIf and give an example of it ?

Ans:- We use the ngIf directive to show or remove a art of the view based on some expression. If the expression is true, that part of the view is rendered else it is completely removed from the DOM. An example of ngIf is given below.

```
export class AppComponent {
  isVisible: boolean = false;
  toggleView () : void {
    this.isVisible = !this.isVisible;
  }
}
```

Inside the component we set a property called isVisible to set the status to true or false. Here we also have a method called toggleView to change the value of the property isVisible.
Now inside the view we need to bind the value of the property with an element and also set a button to toggle the status of the property when clicked. So our view should be looking something like this.

```
<div>
    <button (click) = 'toggleView()'>Toggle View</button>
    <h1 *ngIf = 'isVisible'>I am bound to the ngIf directive.</h1>
</div>
```

Here if the value of the property isVisible is false, the heading element will be removed from the DOM. If it is true, then it will be visible.

11. What is lazy loading ?

Ans:- Lazy loading is the process of loading some parts or components of an application when required or demanded. In the lazy loading process all the modules are not loaded to the root model so that they don't load when the application first loads. Lazy loading is a very effective method which is particularly useful for applications which are data intensive. This process makes the loading of application much faster.

12. What is ahead of time compilation and its advantages in angular ?

Ans:- Ahead of time compilation (AOT) is one of the two process of code compilation where in the html, css and typescript files get compiled to create build files. To enable AOT we need to set a flag in the angular.json file.
The advantages of AOT are listed below.

- Since by AOT angular takes the build files only the tie to render and start is faster than than the Just In Time compilation
- We can get compilation errors in the build time, so that our code does not break when deployed on the server.
- Since the application is already built we can achieve faster download.

13. What is the importance of polyfills.ts file in angular ?

Ans:- Browser compatibility is achieved in angular framework by the polyfills.ts file. If we take a close look at the polyfills.ts file we can see several libraries imported in it to support javascript native functions in all the browsers. So while creating our application we need to take care of which packages to import and use in the polyfills.ts file so as to make them compatible in different browsers.

14. How to improve the performance of an angular application ?

Ans:- We can take some of the following steps to better the performance of our angular application.

- We need to always use the AOT mode of compilation
- We need to employ lazy loading to make it load faster.
- Need to reduce the number of bindings in the view.
- Need to reduce the use of unnecessary imports of package
- Need to avoid external libraries like jQuery to reduce the loading time.

15. What is the difference between constructor and ngOnInit methods ?

Ans:- Following are some of the differences between the constructor and ngOnInit methods**.**

- The constructor method is a typescript feature whereas ngOnInit is an angular life cycle hook.
- The constructor method is the first method called during instantiation of the component or service and ngOnInit is called after all the bindings have been established.
- Constructor is the first function to be executed while ngOnInit is executed later in the life cycle.

16. What are the different life cycle stages of an angular component ?

Ans:- Following are the life cycle stages of an angular component.

- ngOnChanges - called when an input binding value changes.
- ngOnInit - after the first ngOnChanges.
- ngDoCheck - after every run of change detection.
- ngAfterContentInit - after component content initialized.
- ngAfterContentChecked - after every check of component content.
- ngAfterViewInit - after component's view(s) are initialized.
- ngAfterViewChecked - after every check of a component's view(s).
- ngOnDestroy - just before the component is destroyed.

17. What are the differences between the component and directive ?

Ans:- Following are the main differences between a component and a directive.

- @Component decorator is used for component and @Directive decorator is used for the directives.
- Components have a view associated with them whereas Directives do not have a view of their own.
- Components are used as a part of the entire view but directives are used to add behavior to an existing DOM element.
- There can be only one component selector for representing one DOM node while we can have multiple directives associated with one single DOM node.

18. What is the @Input decorator in angular ?

Ans:- The @Input decorator is used to pass data from the parent component to a child component. We set a roperty in the parent component and access this property in the child component by means of @Input decorator.
An example of implementation of the @Input decorator is given below.

Parent Component view:-

```
<div>
    <app-child [parentData]="data"></app-child>
</div>
```

Here we set the element for the child component and set an attribute called parent data which is to be passed to the child component. The data property is a property of the parent component which is to be passed to the child component.

Parent Component:-

```
export class AppComponent {
  data: number = 10;
}
```

Here we set the property some value so that it can be passed to the child component.

Child Component:-

```
export class ChildComponent {
  @Input() parentData: number;

  constructor() { }
}
```

Here we accept the data coming from the parent component by the @Input decorator.

Child Component View:-

```
<p>
  {{parentData}}
</p>
```

here we bind the incoming data from the parent component to the view of the child component.

19. What is the difference between promise and observable ?

Ans:- Following are the differences between a promise and an observable.
A promise is called once and it returns a single value whereas an observable is subscribed and it sends a stream of values whenever there is a change in value at the source.
A promise is not cancellable whereas an observable is cancellable.

20. What is a pipe in angular ?

Ans:- Pipes in angular are used to format data to visualize on the UI. Some of the in-built pipes in angular are given below.

- Datepipe
- UpperCasepipe
- lowerCasepipe
- Currencypipe
- Percentpipe
- jsonpipe
- Asyncpipe

By functionality there are two kinds of pipes.

- Pure pipe
- Impure pipe

21. What is @Inject ?

Ans:- @Inject is a decorator which is used to inject dependencies manually in angular. It is a manual mechanism of letting angular know that a parameter must be injected.
Let's see it in action for a component.

```
import { Component, Inject } from '@angular/core';
import { CustomsvcService } from './customsvc.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  data: number = 10;
  constructor (@Inject(CustomsvcService) private customsvc : CustomsvcService) {
    this.customsvc.logger('Satyapriya Mishra');
  }
}
```

Here we can see that we are injecting a service CustomsvcService with the @Inject decorator. Here we are just concerned about the type of the service instead of instantiating it. But in most of the cases we need not be bothered about injecting it manually by @Inject decorator as internally typescript takes care of this thing. So in a normal scenario we will we will be injecting the above service in the following manner.

```
import { Component, Inject } from '@angular/core';
import { CustomsvcService } from './customsvc.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  data: number = 10;
  constructor (private customsvc : CustomsvcService) {
    this.customsvc.logger('Satyapriya Mishra');
  }
}
```

22. What is the significance of @Injectable ?

Ans:- @Injectable is a decorator which is normally used by services in angular. In any of our service even if we don't use the @injectable decorator, it will work properly fine provided that the service does not have any dependency on any other service or module. In that case the application will not work saying "can't resolve all parameters of service (?)".

Let's suppose we have a very simple class like the below code snippet which does not have the @Injectable() decorator.

```
export class CustomsvcService {

  constructor() { }
}
```

Now when this class is getting transpiled into javascript the compiled code will be like below.

```
__webpack_require__.r(__webpack_exports__);
/* harmony export (binding) */ __webpack_require__.d(__webpack_exports__, "CustomsvcService", function() { return CustomsvcService; });
var CustomsvcService = /** @class */ (function () {
    function CustomsvcService() {
    }
    return CustomsvcService;
}());
```

Now we will add a @Injectable() decorator to the typescript class like this.

```
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})
export class CustomsvcService {

  constructor() { }
}
```

Now if we generate the transpiled code of this class we can have the following output.

```
var CustomsvcService = /** @class */ (function () {
    function CustomsvcService() {
    }
    CustomsvcService = __decorate([
        Object(_angular_core__WEBPACK_IMPORTED_MODULE_0__["Injectable"])({
            providedIn: 'root'
        }),
        __metadata("design:paramtypes", [])
    ], CustomsvcService);
    return CustomsvcService;
}());
```

If we look at the highlighted part we can see that by using the @Injectable() decorator we have some additional metadata generated which is taking care of the parameters for this service. And when we say parameters to a service we mean the parameters to the constructor of the service which is in other words the dependencies of the service. So by this we can safely conclude that without the @injectable() decorator we can't create a service which itself has a dependency on other service or module.
We can also have a custom decorator for this purpose and we can use this decorator in place of @Injectable().

23. What is zone in angular ?

Ans:- Zone refers to the context in which angular performs change detection for asynchronous operations. Asynchronous operations in angular can happen by three modes.

- Events
- Http requests
- Global APIs such as setTimeout(), setInterval() etc

Whenever any of the above things happen inside the zone of angular an event called **onTurnDone** is triggered and angular initiates a dirty checking of the application. This entire process happens with the aid of a JavaScript library called zone.js.

24. What are forRoot and forChild in angular ?

Ans:- Both forRoot and forChild are used for configuring shared modules**.**

- RouterModule.forRoot - forRoot creates a module that contains all the directives, the given routes, and the router service itself.
- RouterModule.forChild - forChild creates a module that contains all the directives and the given routes, but does not include the router service.

forChild is mainly used for shared modules which are lazily loaded.

25. What is the difference between renderer and elementRef ?

Ans:- The main difference between renderer and elementRef is that renderer acts on the DOM element for manipulation whereas elementRef is a reference to the element inside the DOM which is to be manipulated. A very simple example of the implementation of renderer and elementRef is given below.

```
import { OnInit, ElementRef, Renderer } from "@angular/core";

export class SomeDirective implements OnInit {

  constructor(private _elRef: ElementRef, private _renderer: Renderer){}

  ngOnInit() {
      this._renderer.setElementStyle(this._elRef, 'background-color', 'green');
  }
}
```

Here we can see that we implement a change in style of the DOM element by a method of the renderer class and we select the element with the aid of the elementRef class.

1. What is typescript ?

Ans:- Typescript is a programming language which is considered as the superset of JavaScript. All valid JavaScript code is valid typescript. Typescript code is **transpiled** into JavaScript code by the typescript compiler. Transpiling is the process in which one programming language compiles into another programming language of same level of abstraction.

2. What are the most important language features of typescript ?

Ans:- The most important language features of typescript are listed below.

- TypeScript supports new ECMAScript standards and compiles them to (older) ECMAScript targets of your choosing. This means that you can use features of ES2015 and beyond, like modules, lambda functions, classes, the spread operator, destructuring etc.
- JavaScript code is valid TypeScript code; TypeScript is a superset of JavaScript.
- TypeScript adds type support to JavaScript. The type system of TypeScript is relatively rich and includes: interfaces, enums, hybrid types, generics, union and intersection types, access modifiers and much more. TypeScript makes typing a bit easier and a lot less explicit by the usage of type inference.
- The development experience with TypeScript is a great improvement over JavaScript. The IDE is informed in real-time by the TypeScript compiler on its rich type information.
- To use TypeScript you need a build process to compile to JavaScript code. The TypeScript compiler can inline source map information in the generated .js files or create separate .map files. This makes it possible for you to set breakpoints and inspect variables during runtime directly on your TypeScript code.

3. What is generic data type in typescript ?

Ans:- Generic data type is a very unique feature of typescript in which the data type is set at the run time instead of the compilation time. While declaring the generic data type we use <T> as the default data type. At the run time we set the data type of our variables. By generics we can create components and classes which can be reused with multiple data types without any restrictions. An example of generic data type is given below.

```
class Generics<T> {
    public arr = [];
    addItem (item: T) {
        this.arr.push(item);
    }
}

const g1 = new Generics<number>();
g1.addItem(1); // Works as expected
g1.addItem('string'); // Throws an error as we set the type to number
```

Here we can see that since in the run time we set the data type as number, we can't use the addItem method by passing a string as a parameter. We can use any generic type in typescript. So <U> and <R> are also valid generic types.

4. What object oriented programming features does typescript supports ?

Ans:- Typescript supports all features of object oriented programming namely:

- Abstraction
- Polymorphism
- Inheritance
- Encapsulation

5. How to set a constant inside a class ?

Ans:- We can't set a constant property inside a class, otherwise the typescript compiler will throw an error. Here the thing becomes a bit tricky.

```
class MyClass<T> {
    const property = 1;
}
```

So instead of declaring a constant property with the const keyword we can declare a property to be readonly like the following example.

```
class MyClass<T> {
    readonly property: number = 1;
    changeProperty (val: number) {
        this.property = val;
    }
}
var myObj = new MyClass();
myObj.changeProperty(1);
```

Here in the highlighted part we can see that we get an error while trying to change the value of the readonly property. Thus we confirm that the property behaves more like a constant.

6. What is a .map file ?

Ans:-.map files are source map files that let tools map between the emitted JavaScript code and the TypeScript source files that created it. Many debuggers (e.g. Visual Studio or Chrome's dev tools) can consume these files so you can debug the TypeScript file instead of the JavaScript file.

7. What are getters and setters in typescript ?

Ans:- getter and setter methods in typescript are used to get or set the properties of a class respectively. Let's look at an example to understand this concept better.

```
class boo {
    private _bar:boolean = false;
    get bar():boolean {
        return this._bar;
    }
    set bar(theBar:boolean) {
        this._bar = theBar;
    }
}
```

If we compile the above code we will get the below JavaScript code as output.

```
var boo = /** @class */ (function () {
    function boo() {
        this._bar = false;
    }
    Object.defineProperty(boo.prototype, "bar", {
        get: function () {
            return this._bar;
        },
        set: function (theBar) {
            this._bar = theBar;
        },
        enumerable: true,
        configurable: true
    });
    return boo;
}());
```

Here we can see that the compiled JavaScript also creates two methods get and set as a prototyped property attached to the boo object.

Now to use the getter and setter methods our code has to be updated like this.

```
var myFoo = new boo();
if(myFoo.bar) {          // calls the getter
    myFoo.bar = false;   // calls the setter and passes false
}
```

8. What is the significance of the tsconfig.json file ?

Ans:- This file is used to indicate that directory is a root of Typescript project. This file specifies that root files and compiler options are required to compile that particular project. This file can also be used to streamline the building of the project. Below sample can be taken as an example:

```
{
"compilerOptions": {
"removeComments": true,
"sourceMap": true
},
"files": [
"main.ts",
"othermodule.ts"
] }
```

9. What are decorators in typescript ?

Ans:- Decorators can be used to modify the behavior of a class or become even more powerful when integrated into a framework. For instance, if your framework has methods with restricted access requirements (just for admin), it would be easy to write an @admin method decorator to deny access to non-administrative users, or an @owner decorator to only allow the owner of an object the ability to modify it. Example is given below.

10. How to access a class from outside of it parent module ?

Ans:- To access a class from outside of the module we need to export the class. The following example demonstrates this.

```
module Vehicle {
    export class Car {
        constructor(
            public make: string,
            public model: string) { }
    }
}
const myCar = new Vehicle.Car("Ford", "Figo");
```

In the above example if we try to access the Vehicle class without the export keyword, the compiler will throw us an error stating that property Car does not exist.

1. What is RESTful ?

Ans:- REST stands for Representational State transfer. RESTFUL is referred for web services written by applying REST architectural concept are called RESTful services, it focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language. In RESTFUL web service HTTP methods like GET, POST, PUT and DELETE can be used to perform CRUD operations.

2. What are the features of REST ?

Ans:- Following are the most important features of REST API.

- HTTP for client server communication
- XML/JSON as formatting language
- Simple URI as the address for the services
- Stateless communication

3. What are the most important methods supported by REST ?

Ans:- Following are the most frequently used methods of REST API .

- **GET:** It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.
- **POST:** It submits information to the service for processing; it should typically return the modified or new resource
- **PUT:** At the request URL it update the resource
- **DELETE:** At the request URL it removes the resource
- **OPTIONS:** It indicates which techniques are supported
- **HEAD:** About the request URL it returns meta information

4. What is the difference between PUT and POST ?

Ans:- Following are the most important differences between PUT and POST.

- "PUT" puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes one.
- POST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resource.
- PUT is idempotent meaning, invoking it any number of times will not have an impact on resources.
- However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources

5. What are the most important components of HTTP request ?

Ans:- Following are the most important components of any RESTful api.

- **Verb** – Indicate HTTP methods such as GET, POST, DELETE, PUT etc.
- **URI** – Uniform Resource Identifier (URI) to identify the resource on server.
- **HTTP Version** – Indicate HTTP version, for example HTTP v1.1 .
- **Request Header** – Contains metadata for the HTTP Request message as key-value pairs. For example, client (or browser) type, format supported by client, format of message body, cache settings etc.
- **Request Body** – Message content or Resource representation.

6. What are the most common protocols followed by REST ?

Ans:- RESTful API mainly follows two protocols for exchanging data between client and server.

- HTTP :- Hypertext Transfer Protocol
- HTTPS **:-** Hypertext Transfer Protocol Secured.

7.How many types of HTTP methods are there ?

Ans:- According to the state of the resource there are two kinds of HTTP methods.

- Idempotent:- In this category , multiple REST calls will not affect the source. e.g- GET, PUT
- Non-Idempotent:- Here by making multiple REST calls, the source is altered. e.g-POST

8. What are the major components of HTTP response ?

Ans:- Following are the major components of HTTP response.

- **Status/Response Code** − Indicate Server status for the requested resource. For example 404 means resource not found and 200 means response is ok.
- **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
- **Response Header** − Contains metadata for the HTTP Response message as key-value pairs. For example, content length, content type, response date, server type etc.
- **Response Body** − Response message content or Resource representation.

9. What is statelessness in REST and what are its advantages?

Ans:- As per REST architecture, a RESTful web service should not keep a client state on server. This restriction is called statelessness. It is responsibility of the client to pass its context to server and then server can store this context to process client's further request. For example, session maintained by server is identified by session identifier passed by the client.

10. What is URI and why it is necessary ?

Ans:- URI stands for Uniform Resource Identifier.
It contains the address or location of the resources without which the REST API will not be able trace resources. It is the most primary requirement for making any RESTful operation.