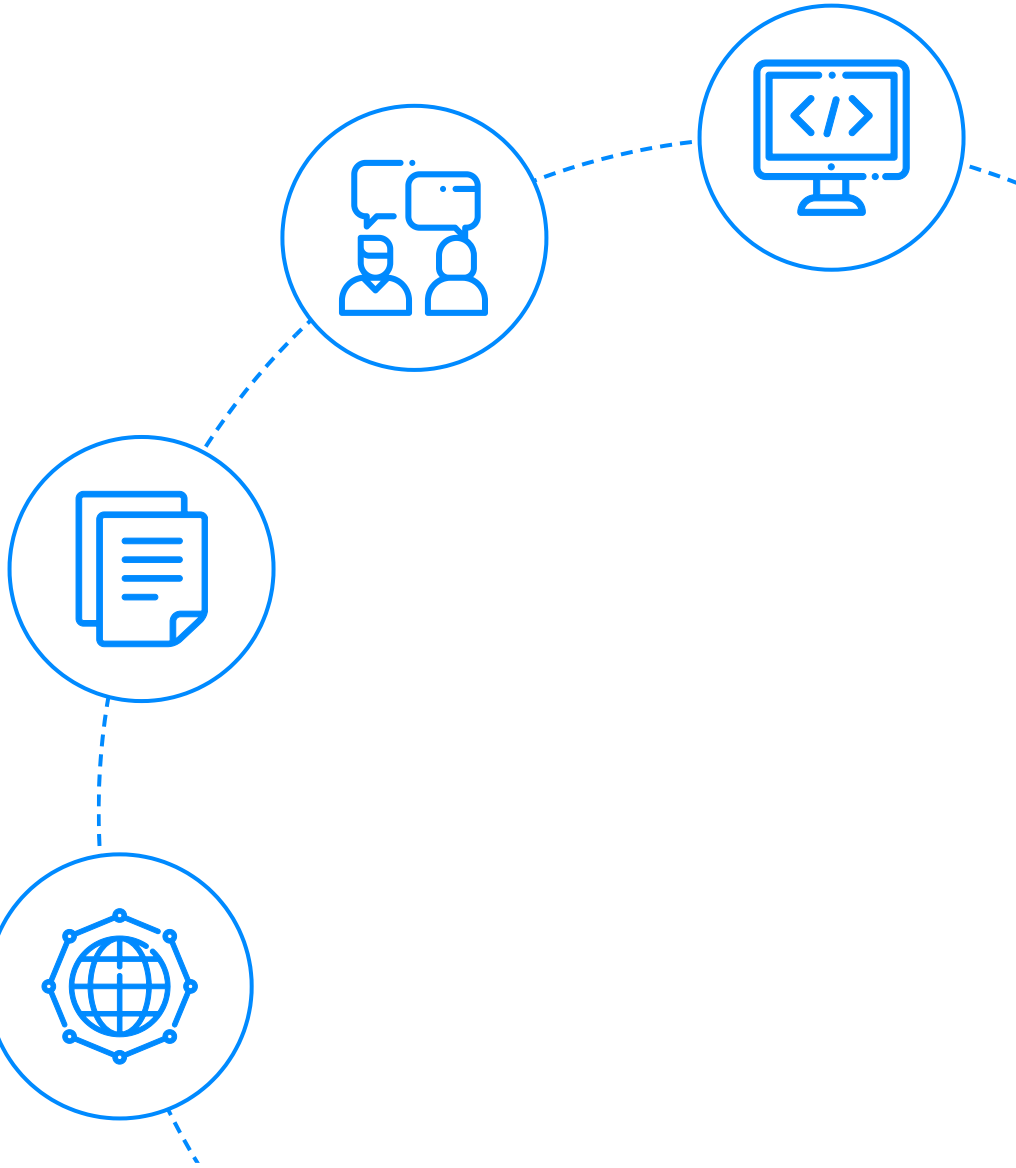
InterviewBit

Full Stack Developer Interview Questions



To view the live version of the page, click here.

Contents

Full Stack Developer Interview Questions for Freshers

Full Stack Developer Interview Questions for Experienced

Full Stack Developer Interview Questions for Experienced

(.....Continued)

19. What do you mean by referential transparency in functional programming?
20. State difference between normalization and denormalization.
21. In Java, what is a connection leak? How can you fix this?
22. What is Promise and explain its states?
23. State the difference between GET and POST.
24. Explain the Restful API and write its usage.
25. What do you mean by MEAN Stack?
26. Do you know how to prevent a bot from scraping your publicly accessible API?
27. What makes MVC (Model View Controller) different from MVP (Model View Presenter)?
28. What do you mean by Temporal Dead Zone in ES6?
29. Why should arrow functions not be used in ES6?
30. What is event bubbling and capturing in JavaScript?
31. Tell me about a project that you worked on and the technologies you used. Why did you choose them?
32. In the past, what was the best implementation or debugging you did?

Frequently Asked Questions

33. How to be a full stack developer?
34. How many types of full-stack developers are there?
35. Which full stack is best?
36. Is full-stack development a good career?

Frequently Asked Questions

(.....Continued)

37. How much do full stack developers earn?

Let's get Started

What is it like to go into the interview process for a Full-Stack Developer role? It is quite common to feel anxious at first, especially if this is your first time on the hot seat. One of the most popular and hottest professions in the tech industry is a Full Stack Development role.

Full-stack developers do not only specialize in front-end and back-end development but are also experts with a wide range of skill sets and knowledge. Its high level of responsibility makes it one of the most important careers in tech and companies are willing to pay handsomely for those with the right skillset and diverse knowledge.

Full Stack Developer Interview Questions and **Answers**

InterviewBit

Likewise, if you're thinking about a career in Full Stack Development, the future is yours. The InterviewBit team has put together a list of 30+ full stack developer interview questions and answers that will help you prepare for this role. These questions have been hand-picked based on what you will likely encounter in the interview.

Before we begin, let's first determine who exactly is a full-stack developer.

Who is a Full-Stack Developer?

A **full-stack web developer** is a person who can develop both client and server software. Simply put, full-stack developers comprehend all the technologies that go into making a website. The responsibilities of Front-end Developers include developing and designing front-end (client-side) web architecture, server-side (back-end) applications, working alongside graphic designers to create web design elements, developing APIs and RESTful services, testing, and debugging software, ensuring cross-platform compatibility and optimization, etc. In addition to mastering HTML and CSS, he/she also knows how to:

- Program a browser (like using JavaScript, jQuery, Angular, or Vue)
- Program a server (like using PHP, ASP, Python, or Node)
- Program a database (like using SQL, SQLite, or MongoDB)

Now let's look at the most common Full Stack Developer interview questions for both freshers and experienced candidates.

Full Stack Developer Interview Questions for Freshers

1. To develop a project from scratch, what technologies and languages would you need or what skills a full stack developer should have?

A **full-stack developer** must be familiar with the following:

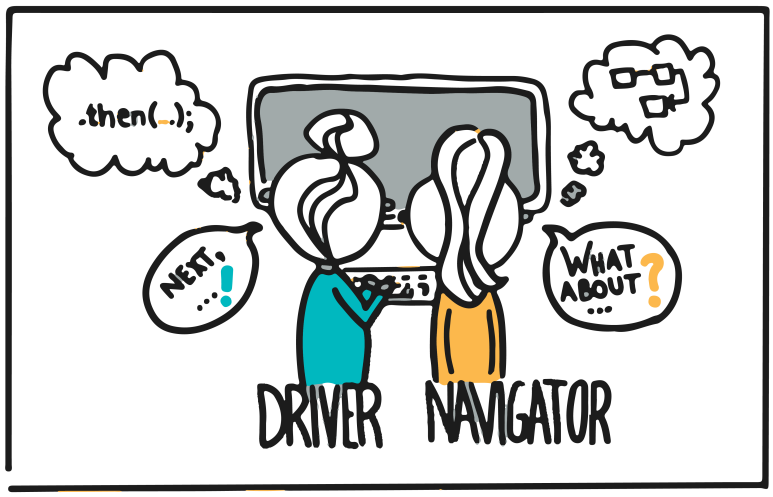
- **Programming Languages:** Full-stack developers should be competent in more than one programming language, such as Java, Python, PHP, Ruby, C++, etc.
- **Front End technologies:** You should be familiar with front-end technologies such as HTML5, CSS3, JavaScript, etc. Knowledge of third-party libraries such as jQuery, Ajax, Angular, ReactJS, etc., is also important.
- **Frameworks:** This requires proficiency in a variety of development frameworks, including Spring, Spring Boot, Django, MyBatis, PHP, Hibernate, and others.
- **Databases and caches:** As a full stack developer, you should also know about various DBMS (Database Management Systems) technologies like MySQL, SQLite, Oracle, and MongoDB. It is helpful to have a basic understanding of caching mechanisms like redis, memcached, varnish.
- **Design Ability:** Having a good understanding of design is also important if you wish to become a successful Full-Stack web developer. Moreover, the person should be aware of the principles of prototyping and UX/UI design.
- **Server:** Experience with Apache or NGINX would be beneficial. Good knowledge of Linux helps enormously when administering servers.

2. Which language is the most preferred by full-stack developers?

Full Stack Developers utilize several programming languages. Ideally, a candidate should be fluent in several languages, preferably some for designing the front end and others for fixing the back end. Since Full Stack developers work with a variety of technologies and applications, they must be proficient in at least two to three of the most popular languages such as **Java, Python, Ruby, PHP, C++, etc**.

3. Explain Pair Programming.

In pair programming, two programmers share only one machine and work together. During the development process, one programmer will be the driver who codes and another will act as the observer (navigator) who will make sure the code is written correctly, proofread and spell-check it, while also deciding where to go next. Roles can be swapped at any time: the driver becomes the observer and vice versa. You can also call it "pairing", "paired programming", or "programming in pairs".

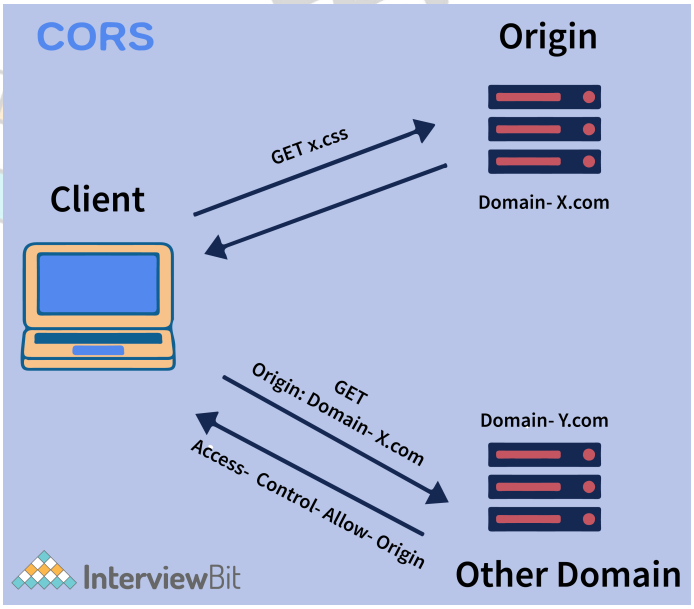


Advantages:

- Two of them will fix the problem if the driver encounters any issues with the code.
- The two programmers working on the same program are only 15% slower than if they worked independently instead of 50%.
- As a result of having another programmer look over your work, you are more likely to write better code. One study showed that it results in 15% fewer bugs than code written by a single programmer.
- It leads to improved collaboration, higher quality, better code, and sustained improved development practices.
- During the project, each person teaches the other, resulting in more efficient and expedited knowledge exchange.
- The team develops better communication skills.

4. What do you mean by CORS (Cross-Origin Resource Sharing)?

CORS refers to cross-origin resource sharing. It's a browser mechanism that allows web pages in one domain to have controlled access to resources in other domains (cross-domain requests). This allows scripts that are run on a browser client to interact with and access resources from other origins. In doing so, it extends and provides greater flexibility to the SOP (Same-Origin Policy). Same-origin policies restrict the ability of a website to access resources outside of its source domain. For example, a JavaScript app that wants to make a call to an API (Application Programming Interface) that runs on another domain will be blocked due to the SOP. A CORS policy was implemented to circumvent restrictions caused by same-origin policies.



In addition, if a website's CORS policy is not configured properly, it may be vulnerable to cross-domain attacks. This means that it cannot stop cross-origin attacks like CSRF (Cross-Site Request Forgery).

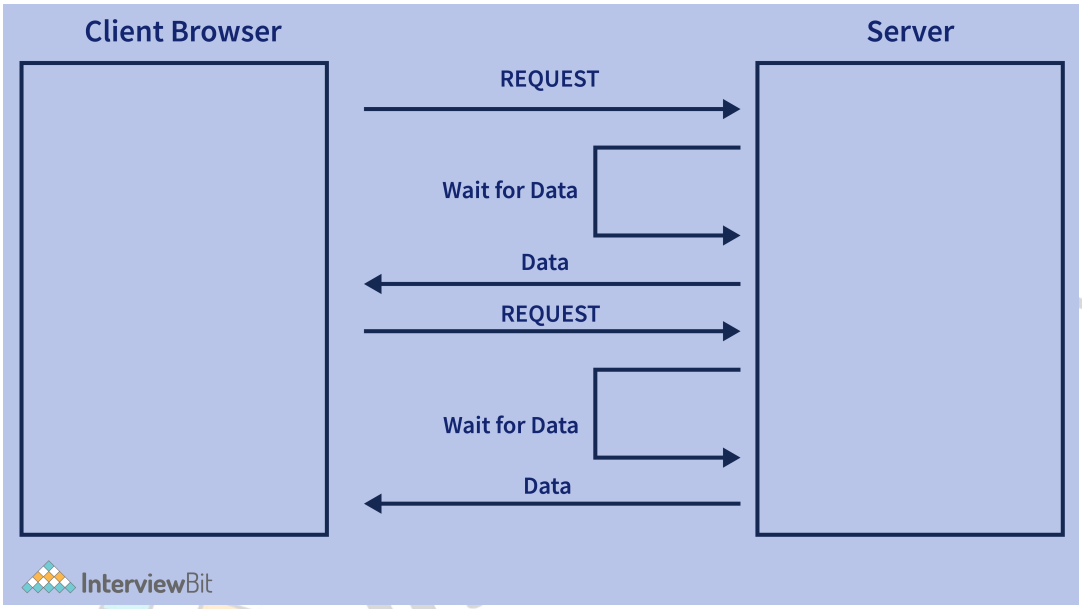
5. What is Callback Hell?

Callback Hell, or Pyramid of Doom, is a common anti-pattern seen in asynchronous programming code (multiple functions running at the same time). This slang term describes a large number of nested "if" statements or functions. In simple terms, Callback hell is a situation where you have multiple asynchronous functions. Those functions depend on one another, so it could get quite messy with so many callback functions nested in so many layers. The use of callback functions leaves you with code that is difficult to read and maintain, and looks like a pyramid as shown below:

This also makes it more difficult to identify the flow of the application, which is the main obstacle to debugging, which is the reason for the famous name of this problem: Callback Hell.

6. Explain Long Polling.

Long polling is defined as a web application development technique used to push information/data from servers to clients as quickly as possible. When a request is made from the client to the server, long-polling maintains the connection between the two. This connection is maintained until the information is ready to be sent from the server to the client. Once a server receives a request from a client, the connection does not close immediately; the connection is only closed once the server has sent the data back to the client or when a timeout threshold has been reached (connection timeout).



7. Can you tell me what are the latest trends in Full Stack Development? Also, how do you keep yourself updated about the new trends in the industry?

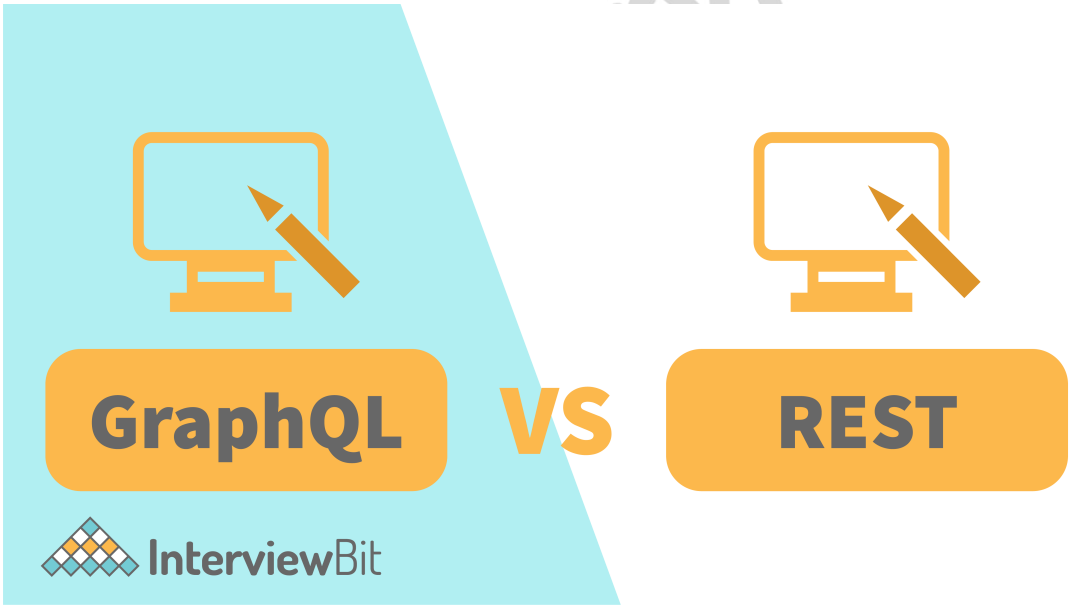
Each business needs emerging technology to thrive, whether it is mobile or web application development. This is why they hire front-end developers, back-end developers, or full-stack developers depending on their technology needs. According to industry experts, candidates who are passionate about full-stack development should be aware of these trends:

- The rise in frameworks and libraries like ReacJs and VueJs, progressive apps, real-time web apps, and mobile web development.
- Enhancements to JavaScript are beneficial to programming
- The emergence of a more compatible extension.

The interviewer may ask you how you keep updated with new trends in the industry. You can explain how you gain your knowledge and understanding by learning from friends, colleagues, or online. That's an appropriate way of answering the question. This is a good time to show off any personal projects you have undertaken that apply your skills. Also, you might mention the webinars or forums that you attend regularly.

8. State difference between GraphQL and REST (Representational State Transfer).

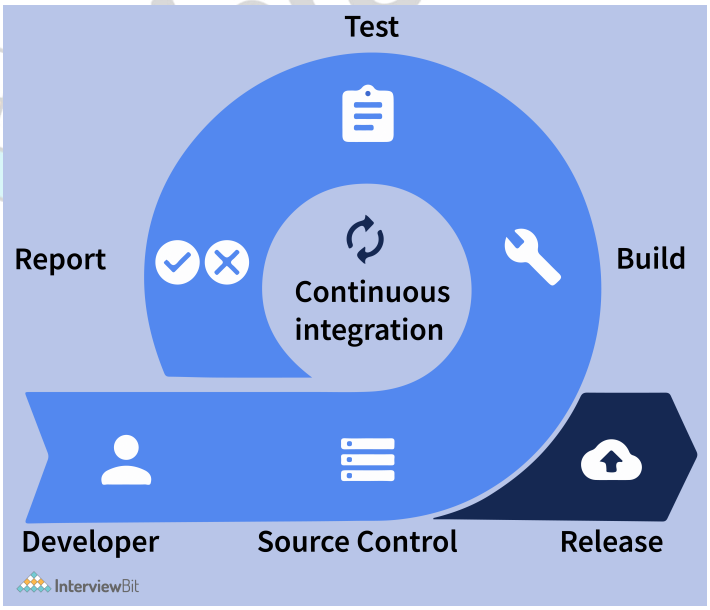
For decades, REST (Representational State Transfer) has been a popular architectural way to design APIs (Application Programming Interfaces), but the popularity of GraphQL over the past few years threatens to topple REST. In general, REST and GraphQL both aim to transfer data over internet protocols including HTTP. How they do so differs greatly, however.



GraphQL	REST
It is a Query Language for APIs that enables declarative data fetching to provide clients control over which data to retrieve from the API.	It is an API design architectural style that defines a set of constraints to create web services.
GraphQL is known for its high predictability. With this, you can send requests to your API and get the exact results you are looking for without having to include anything you don't need. GraphQL queries give predictable results, which improves their usability significantly.	On the other hand, REST's behaviour depends on the HTTP and URI methods used. API consumers can therefore be unsure of what to expect when calling an endpoint.
API security can be ensured by GraphQL, though the security features aren't as mature as those of REST. GraphQL, for instance, assists in integrating data validation, but users are left to figure out how to apply authentication and authorization measures.	API security can be enforced in several ways using REST. You can implement multiple methods of API authentication, such as HTTP authentication, to ensure REST API security.

9. What is CI (Continuous Integration)?

CI (Continuous Integration), as its name implies, is the process of automating and integrating code changes into a single software project, often several times a day. The purpose of this DevOps practice is to enable developers to merge their code changes into a central repository where automated tests and builds can run. Automated tools are used to assert the new code's correctness before integration. A source code version control system is the crux of the CI process. The version control system is also supplemented with other checks like automated code quality tests, syntax style review tools, and more.

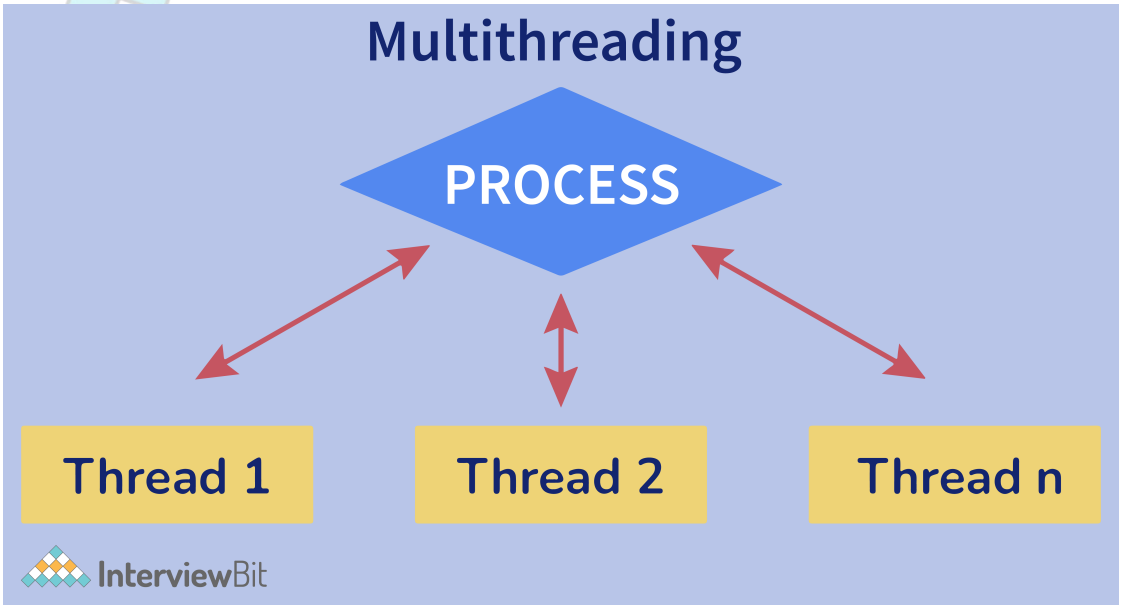


Advantages:

- Integrating regularly has the main benefit of detecting errors quickly and more easily. Since most changes introduced are small, pinpointing the specific change that caused a defect is easy.
- As a result of the smaller code changes and easier fault isolations, CI reduces MTTR (Mean Time to Resolution).
- When CI is incorporated into your organization's development process, you are less likely to have noncritical defects on your backlog. Before production, these small defects are detected and fixed before they are released to the public.

10. Explain the meaning of multithreading.

The thread is an independent part or unit of a process (or an application) that is being executed. Whenever multiple threads execute in a process at the same time, we call this "multithreading". You can think of it as a way for an application to multitask.



Advantages:

- By multithreading, computing resources are also minimized and used more effectively.
- The response time of the application is improved since requests from one thread do not block requests from other threads.
- Consequently, if one of the threads encounters an exception, it will not affect the other threads.
- Multithreading, on the other hand, uses fewer resources than running multiple processes simultaneously.
- The overhead, time usage, and management associated with creating processes are much higher when compared to creating and managing threads.

11. Explain the benefits and drawbacks of using "use strict".

In ECMAScript5, a new feature known as strict mode allows you to run a program or function within a strict operating context. Certain actions, therefore, cannot be taken due to this strict context, and more exceptions are thrown. When the "use strict" statement is used, the browser is instructed to use "strict" mode, which is a more restricted and safer JavaScript feature set. You can specify "use strict" at the top of a function to evaluate the JS in strict mode. In strict mode, more errors are thrown and some features are disabled to make your code more robust, clear, and accurate.

InterviewBit

Advantages:

- Errors are thrown when some common coding mistakes are detected.
- By having strict mode, mistakes that make JavaScript engines tough to optimize can be fixed. Sometimes, strict mode code runs faster than similar, non-strict mode code.
- Generally, it prevents or throws an error when an "unsafe" action is taken (for example, accessing the global object).
- It disables poorly thought-out or confusing features.
- Strict mode simplifies the process of writing "secure" JavaScript.

Disadvantages:

- Many of the functions that most developers use are absent.
- It is not possible to access function.caller or function.arguments anymore.
- If you concatenate scripts written in different strict modes, you may encounter problems.

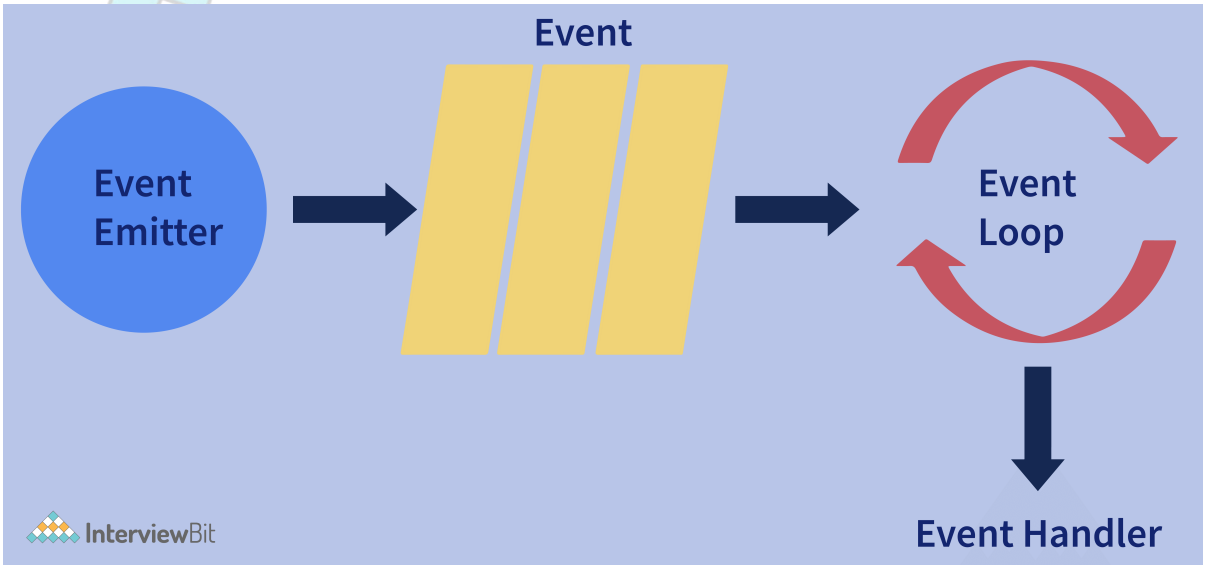
12. What are some of the uses of Docker?

One of the thorniest problems in software development is dealing with the different environments across different machines and platforms. Using Docker, you can isolate your applications from your infrastructure, which is crucial to delivering software quickly. Developers can package and run their applications into containers, executable components that have all the OS (Operating System) libraries and dependencies needed to run the application in any environment. It does not matter what is installed on the host—containers are lightweight and include everything needed to run applications.

- The code has to pass through many different environments as it travels from the developer's machine to production. Consequently, each of these environments may be slightly different. Using Docker streamlines code development and deployment, as it provides a consistent environment from development to production.
- Docker's primary purpose is to simplify configuration. With VM (Virtual Machine), you can run any platform with its configuration on top of your infrastructure. Docker offers the same functionality without the overhead of virtual machines.

13. Explain event loop in Node.js.

The event loop in JavaScript enables asynchronous programming. With JS, every operation takes place on a single thread, but through smart data structures, we can create the illusion of multi-threading. With the Event Loop, any async work is handled by a queue and listener. Consider the diagram below:



Therefore, when an async function (or an I/O) needs to be executed, the main thread relays it to another thread, allowing v8 (Javascript engine) to continue processing or running its code. In the event loop, there are different phases, like pending callbacks, closing callbacks, timers, idle or preparing, polling, and checking, with different FIFO (First-In-First-Out) queues.

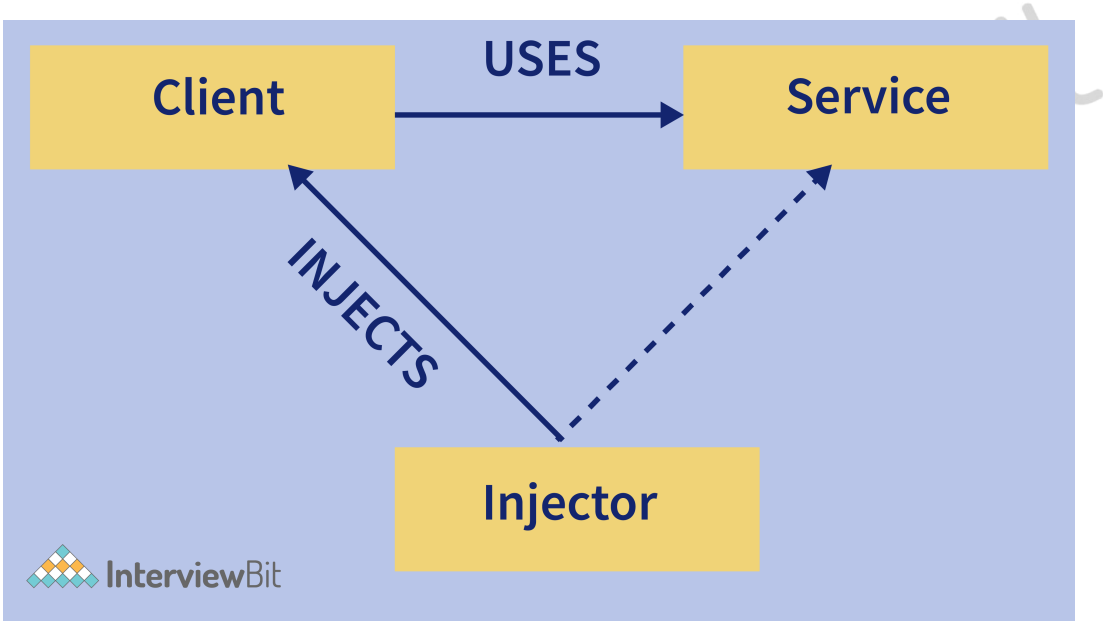
14. Is there a way to decrease the load time of a web application?

Here are some ways to reduce load times for web applications:

- **Image Optimization:** The file size of an image can be dramatically reduced by switching to a different file format. For example, GIFs work well for images with few colors, such as logos, JPEG is ideal for images with lots of colors and details, such as photographs, and PNG format is ideal for transparent images with high quality.
- **Keep JavaScript and CSS in external files:** Embedding JavaScript and CSS in HTML documents forces them to be downloaded every time the HTML document is loaded. In this case, browser caching is not utilized, and the HTML document becomes larger. This is why you should always place CSS and JavaScript in external files; it is a best practice and simplifies maintenance.
- **Reducing redirects:** Too many redirects will delay the loading time of a website. HTTP requests and responses are delayed each time a page redirects. Getting rid of unnecessary redirects on your site will reduce the load time of your site significantly.
- **Load CSS and JavaScript files asynchronously:** Your website contains CSS and JavaScript files that can be loaded either synchronously or asynchronously. As part of synchronous loading, each file is loaded sequentially, in the order it appears on your site. As opposed to synchronous loading, asynchronous loading allows multiple files to be loaded simultaneously, boosting the performance of a website.
- **Minify HTML, CSS, and JavaScript:** If you optimize the way your files load, your pages will load more quickly. You can do the same when it comes to HTML, CSS, and JavaScript code. By eliminating unnecessary spaces, characters, and comments, you can reduce the size of your files. This will make your web pages load faster.

15. Explain dependency injection.

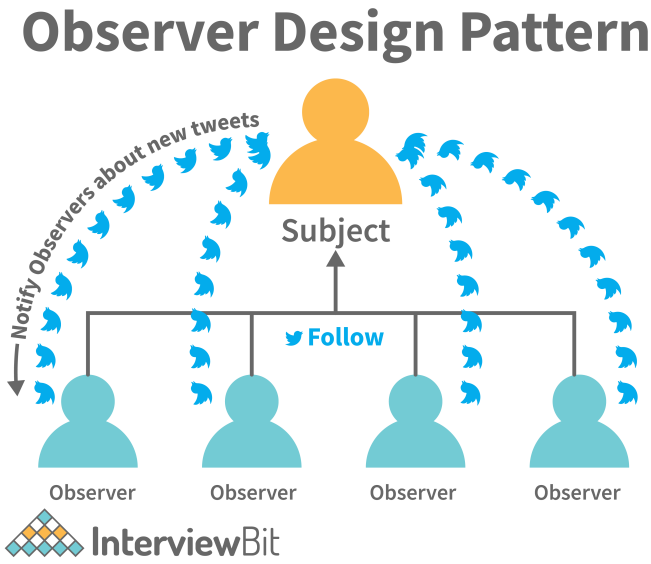
The Dependency Injection (DI) pattern is a design pattern for implementing the Inversion of Control (IoC). Dependent objects can be created outside of classes and made available to classes in different ways. Three types of classes are involved in Dependency Injection as follows:



- Client Class: A client class (dependent class) is one that depends on the service class.
- Service Class: Service (dependency) classes provide services to client classes.
- Injector Class: This class injects the objects from the service class into the client class.

16. What do you mean by observer pattern?

If several objects are tied together in one-to-many relationships, the observer pattern is used. Every time one object is modified, then all of its dependent objects are automatically notified and updated. It falls under the behavioural pattern category. It describes the coupling between the objects and the observer and provides support for broadcast-type communication. The object that observes the state of another object is known as the observer, and the object that is being observed is known as the subject as shown below:



Full Stack Developer Interview Questions for Experienced

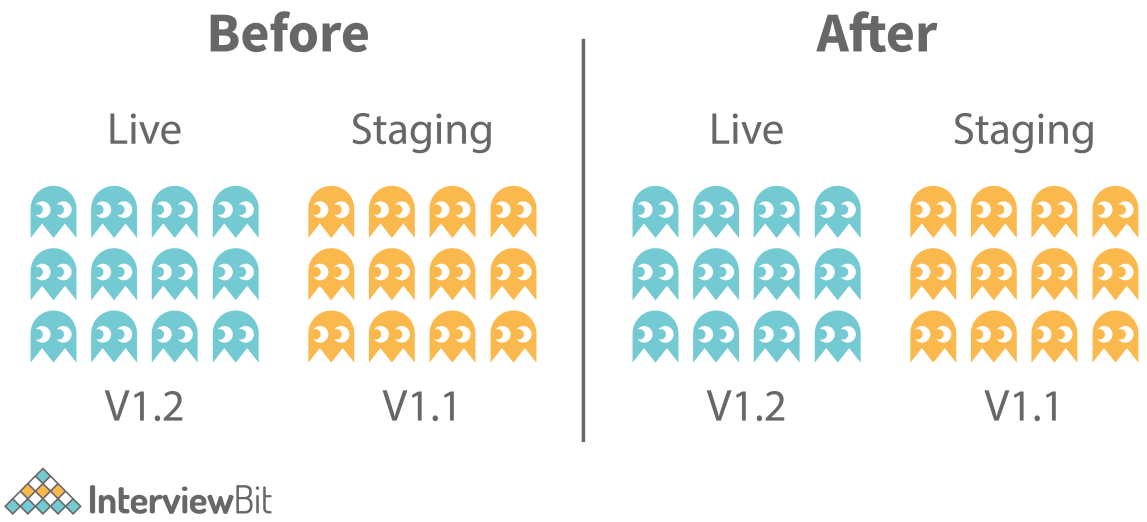
17. State difference between blue/green deployment and rolling deployment.

Today, the software is rapidly created and features are often changed based on customer needs, and then it is deployed into production. Each organization has its unique way of getting new applications into the production environment. Most organizations follow the standard deployment and release strategies such as Blue-Green and Rolling Deployment.

- **Blue-Green Deployment Strategy:**

A deployment strategy like this creates two separate infrastructure environments i.e., blue and green. A blue environment contains older code (old version), while a green environment (production) contains the latest code (new version). There is only one live production environment at any given time.

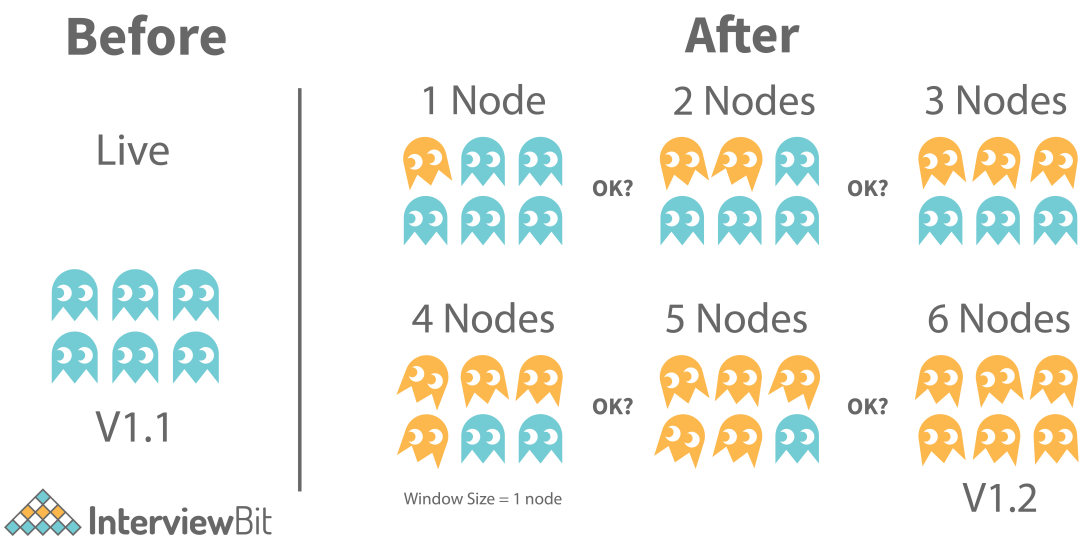
Example: For instance, the green environment is live and is receiving all user traffic, while the clone (blue) is idle. Once a new version of an application is ready for release, it can be deployed to the blue environment for testing. As soon as the new release passes testing, application traffic is switched from green to blue. Blue then becomes the live production environment, and Green becomes idle, ready for testing the next release.



- **Rolling Deployment Strategy**

Using this deployment strategy, old versions of an application are completely replaced with the new versions by completely replacing the infrastructure on which they run.

Example: When a new version must be deployed to all nodes, the new version is deployed to the first node while all other nodes handle end-user traffic. As soon as the new version is successfully installed on the first node, it will begin managing end-user traffic while the new version is being installed on the second node. The process is repeated until all nodes have been successfully upgraded to the new version.



18. Explain inversion of control.

IoC (Inversion of Control), as the name suggests, is a design principle in software engineering. With IoC, different kinds of controls can be inverted in an object-oriented design to attain loose coupling. The term "controls" refers to any other responsibilities a class may have other than its primary responsibility. These include controlling the flow of an application, controlling the creation of objects, or controlling the binding and creation of dependent objects. IoC allows classes to be loosely coupled, making testing and maintenance easier.

19. What do you mean by referential transparency in functional programming?

In functional programming, referential transparency is the key differentiating factor. An expression is considered referential transparent if it can be replaced or substituted with the corresponding value it computes or vice-versa without affecting the program's result.

Example: Imagine that we have an expression called four: `val four= add(1,3)`

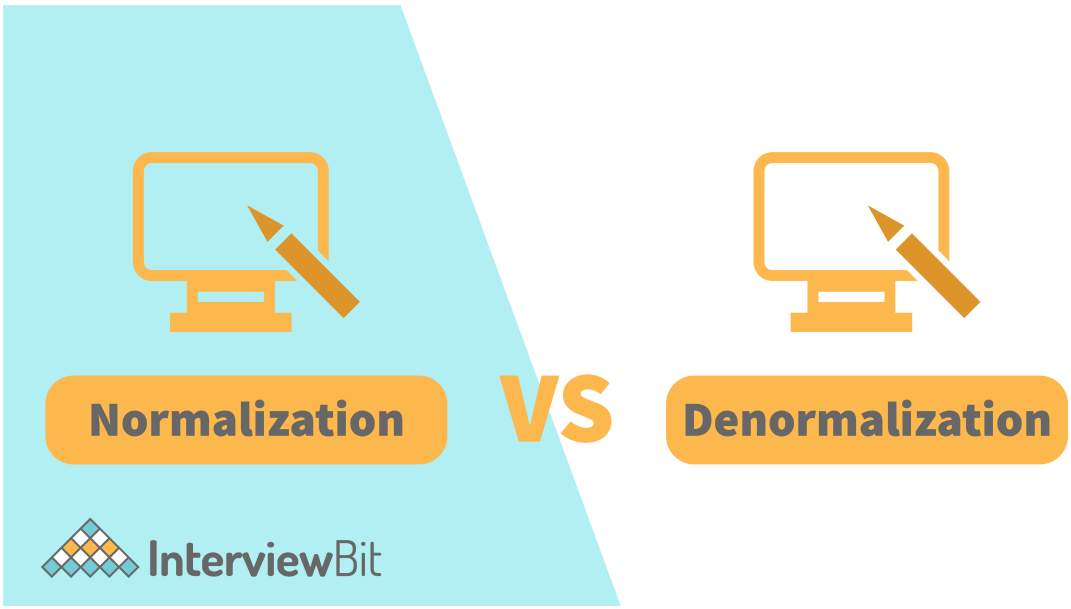
If four is used anywhere in our code, it can safely be replaced with add(1,3), 1 + 3 or 4 wherever it appears. Thus, all the expressions below are equivalent in meaning and output:

```
val eight = four + four
val eight_v2 = add(1,3) + add(1,3)
val eight_v3 = 4 + add(1,3)
val eight_v4 = 8
```

If we can swap back-and-forth between these expressions anytime, anywhere, without altering their meaning or output, then an expression is referentially transparent.

20. State difference between normalization and denormalization.

Normalization and denormalization are the two main methods of altering the structure of a database.



Normalization	Denormalization
Normalization involves removing redundant data (multiple copies of data) from a database and storing consistent, non-redundant data.	It involves combining data from multiple tables into a single so that it can be queried quickly.
It primarily focuses on clearing out unused data and reducing duplicate data and inconsistencies from a database.	On the other hand, denormalization aims to achieve faster query execution by adding data redundancy.
During normalization, tables are reduced in number due to the reduction of data in the database.	Denormalization, on the other hand, involves integrating data into the same database and therefore the number of tables to store the data increases.
Data integrity is maintained by normalization. A change to the data in the table will not impact its relationship with the other table.	Data integrity is not maintained by denormalization.
It optimizes the use of disk space.	It does not optimize disk space.

21. In Java, what is a connection leak? How can you fix this?

If a connection is opened and forgotten about, this is known as a "leak" since each time it occurs, a connection is no longer available for reuse. Connection leaks occur when some database requests or transactions are not closed properly or are not committed, causing the connections to be abandoned and closed permanently.

Java developers commonly experience Connection Leaks when using Connection Pools. In the case where there is a section of code that fails to close a connection properly, a connection will leak from the pool each time the section of code is executed. Eventually, if this situation continues, the pool will run out of connections, which is known as Pool Exhaustion. The application will hang once all available connections have been leaked. We can fix this by closing the connection and paying particular attention to any error handling code.

22. What is Promise and explain its states?

Callback functions are functions that can be passed to another function as arguments and executed there to complete a routine or action. Those functions depend on one another, so it could get quite messy with so many callback functions nested in so many layers. This is what is referred to as callback hell.

As an alternative to callbacks in JavaScript, promises are used to handle asynchronous operations. In addition to handling multiple asynchronous operations, they provide better error handling than callbacks. Promises can be a better way for a user to read the code effectively and efficiently, especially when that particular code performs multiple asynchronous operations. The Promise object represents the result of an asynchronous operation (or its failure) and the resulting value. The promise is in one of the following states:

- **Pending:** In its initial state, neither fulfilled nor rejected.
- **Fulfilled:** Indicating that the operation was successful.
- **Rejected:** Indicating that the operation failed.

23. State the difference between GET and POST.

GET and POST are two different HTTP request methods.

GET	POST
This method is used to request data from a certain resource (via some API URL).	This method is used to send or write data to be processed to a specific resource (via some API URL).
If you use the GET method to send data, the data is added to the URL, and a URL can be up to 2048 characters in length. Therefore, it has restrictions on data length.	It does not impose such limitations.
In comparison to POST, GET is less secure since data is sent as part of the URL. Passwords and other sensitive information should never be sent using GET.	It is a little safer to use POST than GET because the parameters are not saved in the browser history or the web server logs.
Everyone can see the data in the URL.	There is no data displayed in the URL.

24. Explain the Restful API and write its usage.

APIs (Application Programming Interfaces) are sets of rules and protocols that define how software programs or devices can communicate with each other. APIs that conform to the design principles of REST, or representational state transfer, are known as REST APIs. REST APIs may also be referred to as RESTful APIs. Using RESTful APIs, developers can create requests and receive responses via an HTTP request. REST API can also be used for mapping data from a cloud platform to a data warehouse or vice versa.

25. What do you mean by MEAN Stack?

MEAN stands for MongoDB, ExpressJS, AngularJS, and Node.js. It is a collection of JavaScript-based technologies for developing web applications. Despite being a stack of different technologies, all of them are based on the JavaScript language. It is an ideal solution for building dynamic websites and applications as it is a very user-friendly stack. With this free and open-source stack, you can quickly and easily build web-based prototypes.

26. Do you know how to prevent a bot from scraping your publicly accessible API?

As long as the data within the API is accessible to the public, it will technically not be possible to completely prevent data scraping. It is possible, however, to minimize bot activity (automated computer programs on the internet that perform certain tasks) by throttling or rare limiting. Rare limiting will be able to prevent a certain device from making an unlimited number of requests within a defined time. If too many requests are made beyond the defined limit, a 429 Too Many Attempts HTTP error is thrown. It is vital to record more than just the IP address of the device since the IP address is not unique to each device and can stop the whole network from accessing the API.

27. What makes MVC (Model View Controller) different from MVP (Model View Presenter)?

Developers prefer to develop Android applications by utilizing a software architecture pattern. Architecture patterns allow you to express and define a structural schema for software systems. Developers can easily maintain the software and continue to add features to the software in the future. The two most popular android architecture patterns are MVC (Model View Controller) and MVP (Model View Presenter).

MVC	MVP
MVC suggests splitting the code into three components. As soon as the developer creates a class or file for an application, he or she must categorize it into one of three layers: Model, View, and Controller.	This is an architectural pattern that helps compensate for some of the shortcomings of MVC. It is composed of three components i.e., Model, View, and Presenter.
The controller serves as a bridge between the view and model layers and therefore provides the application's user interface. As soon as the Model changes, the Controller updates the View.	The presenter pulls data from the model and applies the UI (user interface) logic to determine what to show. In response to the user's input notification, it manages the state of the View and takes appropriate actions.
Controllers and views have a many-to-one relationship since one Controller can select different Views depending on the operation required.	Presenter and View have a one-to-one relationship since the Presenter class manages only one View at a time.
Support for unit testing is limited.	The unit testing process is highly supported.

28. What do you mean by Temporal Dead Zone in ES6?

Before ES6, variable declarations were only possible using var. With ES6, we got let and const. Both let and const declarations are block-scoped, i.e., they can only be accessed within the " { } " surrounding them. On the other hand, var doesn't have such a restriction. Unlike var, which can be accessed before its declaration, you cannot access the let or const variables until they are initialized with some value. Temporal Dead Zone is the time from the beginning of the execution of a block in which let or const variables are declared until these variables are initialized. If anyone tries to access those variables during that zone, Javascript will always throw a reference error as given below.

```
console.log(varNumber); // undefined
console.log(letNumber); // Throws the reference error letNumber is not defined
var varNumber = 3;
let letNumber = 4;
```

Both let and const variables are in the TDZ from the moment their enclosing scope starts to the moment they are declared.

29. Why should arrow functions not be used in ES6?

One of the most popular features of ES6 is the "arrow functions" (also known as "fat arrow functions"). Arrow functions are a new way to write concise functions. Arrow functions offer a compact alternative to traditional function expressions, but they have limitations and cannot be used in every case.

The following is an ES5 function:

```
function timesTwo(params) {
        return params * 2
     }
timesTwo(5);  // 10
```

The same function can also be expressed as an arrow function in ES6 as follows:

```
var timesTwo = params => params * 2
timesTwo(5);  // 10
```

Differences & Limitations:

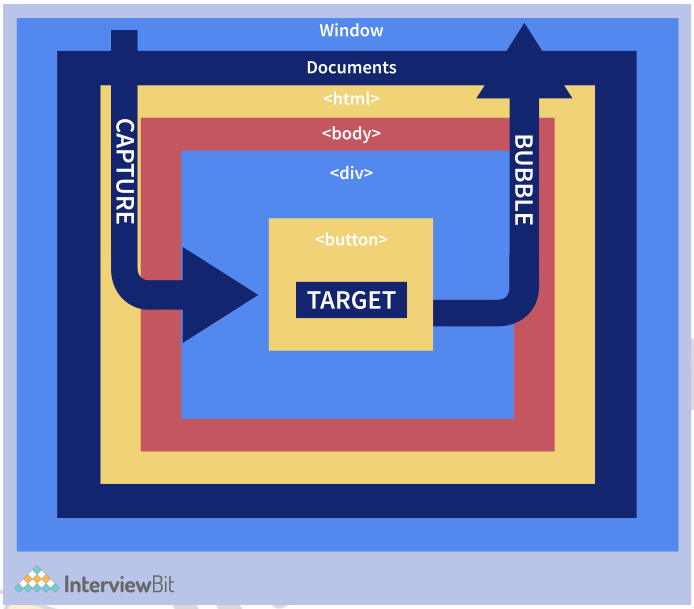
- Has no bindings to 'this' or 'super', so it shouldn't be used as a method.
- Has no new.target keyword.
- The call, apply, and bind methods are not suitable since they rely on establishing scope.
- Not suitable for use as constructors.
- Not possible for an arrow function to use the yield keyword within its body, etc.

30. What is event bubbling and capturing in JavaScript?

The propagation of events inside the DOM (Document Object Model) is known as 'Event Flow' in JavaScript. The event flow defines the order or sequence in which a particular web page receives an event. Accordingly, event flow (propagation) in JS is dependent on the following aspects:

- **Event Bubbling:** With Event Bubbling, the event is captured and handled first by the innermost element, and then propagates to the outermost element. Events propagate up the DOM tree from child elements until the topmost element is handled.
- **Event Capturing:** With Event Capturing, the event is captured and handled first by the outermost element, and then propagates to the innermost element. Event cycles propagate starting with the wrapper elements and ending with the target elements that initiated the event cycle.

The following diagram will help you to understand the event propagation life cycle.



31. Tell me about a project that you worked on and the technologies you used. Why did you choose them?

With this question, the interviewer can figure out the methodology of the full stack web developer and determine if he is sharp and precise in selecting the right toolset.

While explaining the reasons for choosing a particular toolset, you should be as specific as possible. You must demonstrate that you can develop both the front and backend of the web application. Having more experience on one side of the development game than the other is okay, but you should show you are capable of handling both ends of the application.

32. In the past, what was the best implementation or debugging you did?

By asking this question, the hiring manager will get an idea of the type and complexity of past projects you have completed. You should clearly state the problems you encountered and the steps you took to overcome them. In addition, you can talk about the lessons you learned from the issue.

Conclusion

A potential employer doesn't expect applicants to know everything, whether they're freshers or experienced. However, a well-prepared and enthusiastic candidate will undoubtedly stand out. In a full stack developer interview, candidates should be able to demonstrate excellent organizational skills and precision in their work. Also, honesty and transparency pay dividends over the long run.

In this article, we have covered the top 30+ Full-Stack developer interview questions & answers for freshers and experienced candidates so you can succeed in your next full-stack developer interview. Questions like those listed above are quick, insightful, and provide key cues that are vital to the interview process. If you review frequently asked full stack developer interview questions and answers, you increase your chances of getting hired. Make sure you review all the questions and answers.

We hope to have cleared up your doubts and guided you in the right direction. Good luck with your full stack developer interview!

Frequently Asked Questions

33. How to be a full stack developer?

To become a full stack developer, follow these six steps:

- Plan your learning path according to your career goals
- Learn the fundamentals, necessary technologies, and develop the skills you need to be a full-stack developer and necessary technologies.
- Learn from tutorials and videos, but learn to code as well.
- Based on what you learn, create your own project.
- Make a portfolio of your work.
- Start your job search and nail the interview.

34. How many types of full-stack developers are there?

Following are the most common types of full-stack developers based on their specific stacks:

- MEAN Stack (MongoDB-ExpressJS-AngularJS-NodeJS).
- MERN Stack (MongoDB-ExpressJS-ReactJS-NodeJS).
- LAMP Stack (Linux, Apache, MySQL, PHP).
- LEMP Stack (Linux, Nginx, MySQL, PHP).
- Full-Stack Python.
- Full-Stack Elixir.
- Full-Stack Django.
- Full-Stack Java.
- Full-Stack Ruby on Rails.

35. Which full stack is best?

One of the most popular tech stacks is the MEAN stack (MongoDB-ExpressJS-AngularJS-NodeJS). MEAN stack has many applications, including cloud application development. MongoDB, Express.js, AngularJS, and Node.js are among the JavaScript technologies in MEAN used to build web applications.

36. Is full-stack development a good career?

Definitely, it's a great career choice. Having a full-stack developer job is one of the most productive careers you can have, and its high level of responsibility makes it one of the most important careers in tech. Full-stack developers do not only specialize in front-end and back-end development but are also experts with a wide range of skill sets and knowledge. Full Stack Developers are in high demand and companies are willing to pay handsomely for those with the right skillset and diverse knowledge. Here's a Resume Sample.

37. How much do full stack developers earn?

A Full Stack Developer is one of the most highly paid professionals. In India, the average annual salary for a Full Stack Developer is around ₹6,66,697. With 1-4 years of experience, a full-stack developer earns an average of INR 6,53,006-10,00,000. Employees with between 5 and 9 years of experience can expect to earn between INR 12 and 17 lakhs per year. **Learn More**.

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions