Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

Register | Login | Logout | Contact Us

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Pressed for time? Java/JEE Interview FAQs › FAQ JEE Job Interview Q&A Essentials › 01: ♦ 12 Web basics every Java web developer must know

01: ♦ 12 Web basics every Java web developer must know

Posted on August 22, 2014 by Arulkumaran Kumaraswamipillai

Q1. HTTP is a stateless protocol, so how do you maintain state? How do you store user data between requests?
A1. This is a commonly asked interview question. The "http protocol" is a stateless request/response based protocol. You can retain the state information between different page requests as follows:

HTTP Session. A session identifies the requests that originate from the same browser during the period of conversation. All the servlets can share the same session. The JSESSIONID is generated by the server and can be passed to client through cookies, URL re-writing (if cookies are turned off) or built-in SSL mechanism. Care should be taken to minimize size of objects stored in session and objects stored in session should be serializable. In a Java servlet the session can be obtained as follows:

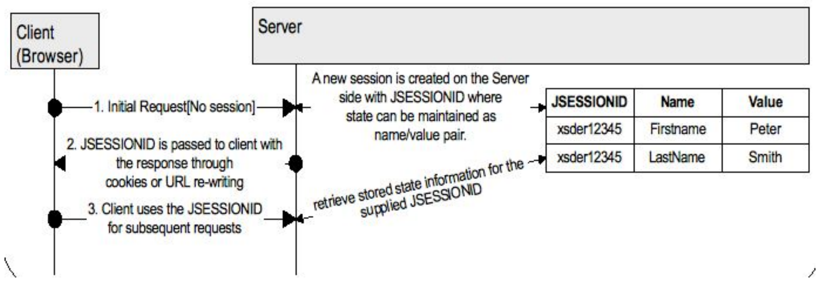
```
HttpSession session = request.getSession(true);

//To put/get a value in/from the session
Name name = new Name(“Peter”);
session.setAttribute(“Firstname”, name); //sessi

session.getAttribute(“Firstname”);//get a value.

//If a session is no longer required e.g. user h
session.invalidate();

```


Client (Browser)
Server
1. Initial Request[No session]
A new session is created on the Server side with JSESSIONID where state can be maintained as name/value pair.
2. JSESSIONID is passed to client with the response through cookies or URL re-writing
3. Client uses the JSESSIONID for subsequent requests
retrieve stored state information for the supplied JSESSIONID
JSESSIONID
Name
Value
xsder12345
Firstname
Peter
xsder12345
LastName
Smith

Web basics

Q. Session tracking uses cookies by default. What would you do if the cookies are turned off?
A. If cookies are turned off, you can still enable session tracking using URL rewriting. This involves including the session ID within the link as the name/value pair as shown below.

```
http://localhost:8080/myWebCtxt/purchase.do;jsess
```

Adding session ID to each and every link is cumbersome and hence is simplified by the following methods: response.encodeURL(givenURL) to associate a session ID with a given URL and if you are using redirection then *response.**encodeRedirectURL**(givenURL)*. When you invoke the method encodeURL(givenURL) with the cookies turned on, then session ID is not appended to the URL. Now turn the cookies off and restart the browser. If you invoke the

⊞ SQL, XML, UML, JSC
⊞ Hadoop & BigData Int
⊞ Java Architecture Inte
⊞ Scala Interview Q&As
⊞ Spring, Hibernate, & I
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A 1
⊞ ▶ Free Java Interviev

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience I
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2

encodeURL(givenURL) with the cookies turned off, the session ID is automatically added to the URL

Hidden Fields on the pages can maintain state and they are not visible on the browser. The server treats both hidden and non-hidden fields the same way.

```
<input name=""Firstname"" type=""hidden"" value="
<input name=""Lastname"" type=""hidden"" value=""

```

The disadvantage of hidden fields is that they may expose sensitive or private information to others.

URL re-writing will append the state information as a query string to the URL. This should not be used to maintain private or sensitive information.

```
Http://MyServer:8080/MyServlet?Firstname=Peter&La
```

Cookies: A cookie is a piece of text that a Web server can store on a user's hard disk. Cookies allow a website to store information on a user's machine and later retrieve it. These pieces of information are stored as name-value pairs. The cookie data moves in the following manner:

– If you type the URL of a website into your browser, your browser sends the request to the Web server. When the browser does this it looks on your machine for a cookie file that URL has set. If it finds it, your browser will send all of the name-value pairs along with the URL. If it does not find a cookie file, it sends no cookie data.

– The URL's Web server receives the cookie data and requests for a page. If name-value pairs are received, the server can use them. If no name-value pairs are received, the server can create a new ID and then sends name-value pairs to your machine in the header for the Web page it sends. Your machine stores the name value pairs on your hard disk.

⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)
⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all
⊞ Can you write code? (
⊞ ♦ Complete the given
⊞ Converting from A to I
⊞ Designing your classe
⊞ Java Data Structures
⊞ Passing the unit tests
⊞ What is wrong with th
⊞ Writing Code Home A
⊞ Written Test Core Jav
⊞ Written Test JEE (1)

How good are your .....?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

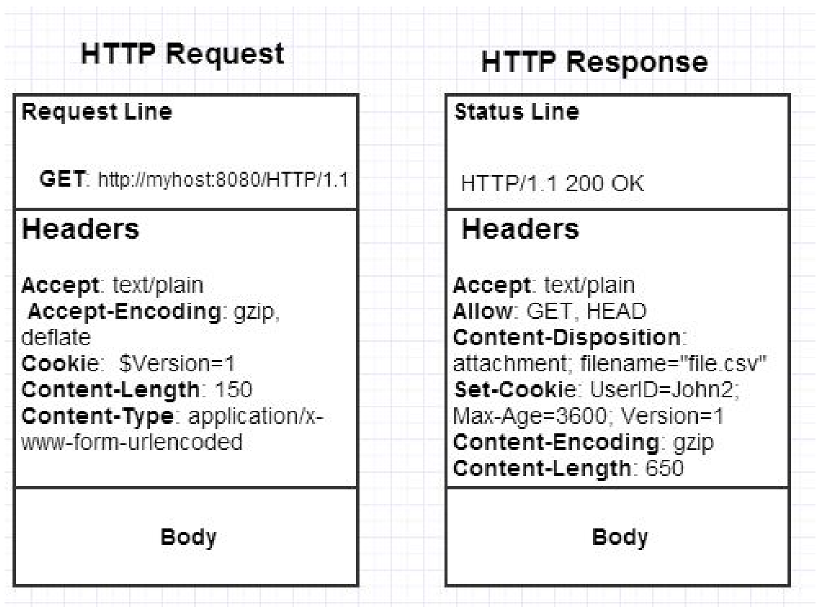
Cookies can be used to determine how many visitors visit your site. It can also determine how many are new versus repeated visitors. The way it does this is by using a database. The first time a visitor arrives; the site creates a new ID in the database and sends the ID as a cookie. The next time the same user comes back, the site can increment a counter associated with that ID in the database and know how many times that visitor returns. The sites can also store user preferences so that site can look different for each visitor.

Q2. Are states bad? if yes, why?
A2. State isn't bad, it's a necessity. But favor stateless architecture for the number of reasons described below.

— You should ALWAYS make your systems not only as simple as possible, but also as scalable as possible. In a clustered or load balanced environment with multiple servers, maintaining of states requires the user to always return to the same server using sticky sessions or else the whole system breaks down. The answer to this problem is using sticky sessions which force the load balancer to send traffic for a given user to the same server every time. The LB is hardly balancing load if it can't decide who gets the traffic. There are other more complicated methods for sharing sessions across multiple servers via true clustering, but getting this right is more complicated and time consuming.
State can also be problematic from a security point of view due to sharing the state between the client and server. There are methods around this problem using encryption and other techniques, but again can complicate your solution.

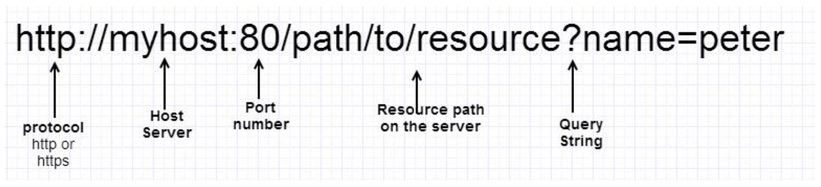
— The single page rich web pages making a number of RESTFul web services via ajax requests is a very popular architecture. Making server side RESTful web services stateless is a central tenant of REST. A truly stateless RESTful web service is not only incredibly flexible and scalable thing, but also takes responsibility for handling security and deciding who can access what in the system. Since there is very little "under the covers stuff" happening to try to make this stuff "easier", you are free to implement security however makes sense for your system.

Q3. What is the structure of an HTTP request and response?
A3. HTTP a stateless protocol, and communication between a host and a client occurs, via a request/response pair. The client initiates an HTTP request message, which is serviced through a HTTP response message in return.



HTTP Request and Response

Q4. What is your understanding of HTTP URLs, HTTP Verbs, and HTTP status codes?
A4. The heart of web communications is the request message, which are sent via Uniform Resource Locators (**URLs**).


http://myhost:80/path/to/resource?name=peter
protocol
http or https
Host Server
Port number
Resource path on the server
Query String

URL

URLs reveal the identity of the particular host with which we want to communicate, but the **action** that should be performed on the host is specified via **HTTP verbs**. A client can perform a number of actions like GET (fetch an existing resource), POST (create a new resource), PUT (update an existing resource,), DELETE (delete an existing resource), etc.

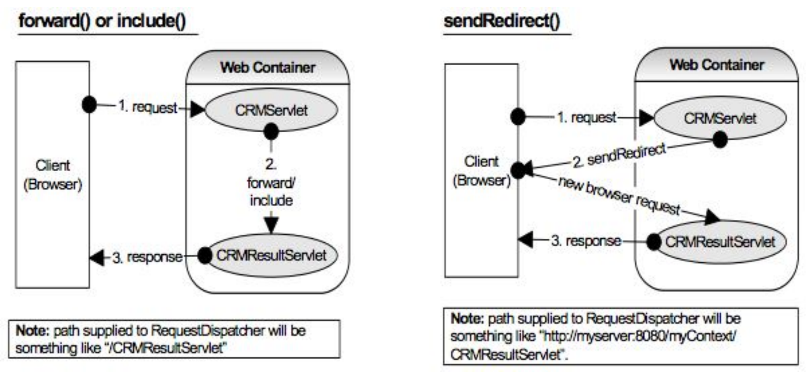
With URLs and verbs, the client can initiate requests to the server. In return, the server responds with **status codes** and message payloads. The status code is important and tells the client how to interpret the server response. he most common code is 200 OK, which tells the client that the request was successfully processed.

3xx like 301, 302, etc indicate **redirection**, which requires the client to take additional action. The most common use-case is to jump to a different URL in order to fetch the resource. 4xx: like 404. etc means **Client Error**. These codes are used when the server thinks that the client is at fault, either by requesting an invalid resource or making a bad request. 5xx: like 501, 503, etc means **Server Error**. This class of codes are used to indicate a server failure while processing the request. The most commonly used error code is **500 Internal Server Error**.

Q5. What is your understanding of the following terms … forward, redirect (aka sendRedirect), and post-back?
A5.

Forward

— a forward or include is performed internally by the servlet on the server side.
— The browser is completely unaware that it has taken place, so its original URL remains intact.
— You can set request attributes from the forwarding request (e.g. Servlet) to the resource to which it is forwarded (e.g. another Servlet or JSP).



forward, redirect and postback

Redirect

— A redirect is a two step process, where the web application instructs the browser to fetch a second URL, which differs from the original.
— A browser reload of the second URL will not repeat the original request, but will rather fetch the second URL. This makes the second resource link book markable.
— A redirect can be marginally slower than a forward as it requires two browser requests as opposed to one with the forward.
— Any attributes placed in the original request scope are not available to the second request, since it is a new request.

Post-back

— The HTTP verb POST is used to send data to the server in the body, with XML, JSON, or form fields.
— The term "back" really means that you retrieved the page initially with a GET verb to show the user the

elements, and at the end you're sending data back. So, a PostBack is a POST request for a page that is not the first request.

Q6. What is exactly is an Ajax request?
A6. Ajax is browser technology, which stands for Asynchronous Javascript and XML, and an Ajax call is an asynchronous request initiated by the browser that does not

directly result in a page transition. An Ajax request is sometimes called an XHR request or "XmlHttpRequest", which is the name most browsers give the object used to send an Ajax request send/receive XML, JSON, plain text or HTML. So, the "X", Ajax was coined to send and receive XML messages asynchrously, but it can send/recive JSON, plain text, etc.

Conventional web application trasmit information to and from the sever using synchronous requests. This means you fill out a form, hit submit, and get directed to a new page with new information from the server. With AJAX when submit is pressed, JavaScript will make a request to the server, interpret the results and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server. So, Ajax is Data-driven as opposed to the conventional page-driven approach.

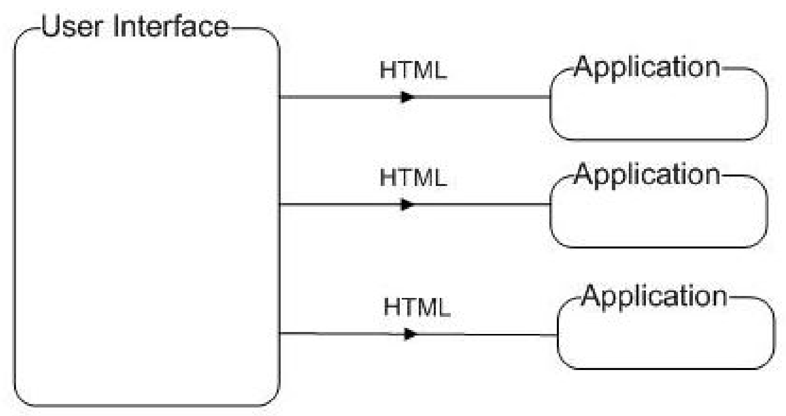
The modern Rich Internet Applications (**RIA**) use the design concept of "**single page web design**", where a single rich page makes Ajax based service calls to render different sections of a page instead of the traditional approach of loading a new page of each user action. The "single page web design" can make use of client side and server side technologies.

Q7. What do you understand by client side and server side templating?
A7. The modern rich single page web applications built today harness the power of dynamically typed and interpreted languages like JavaScript for faster prototyping and increased developer productivity and statically typed Java based frameworks for maintainability, robustness, easier refactoring, and scalability. The modern browsers using faster JavaScript engines and efficient minification of JavaScript files have made client side templating a reality.

The server side templates are such as JSPs, Facelets for JSF, Apache Velocity, tiles, Sitemesh, etc.
The client side templating libraries based on JavaScript include angularjs, backbone, ember, etc

Server side templating and data composition

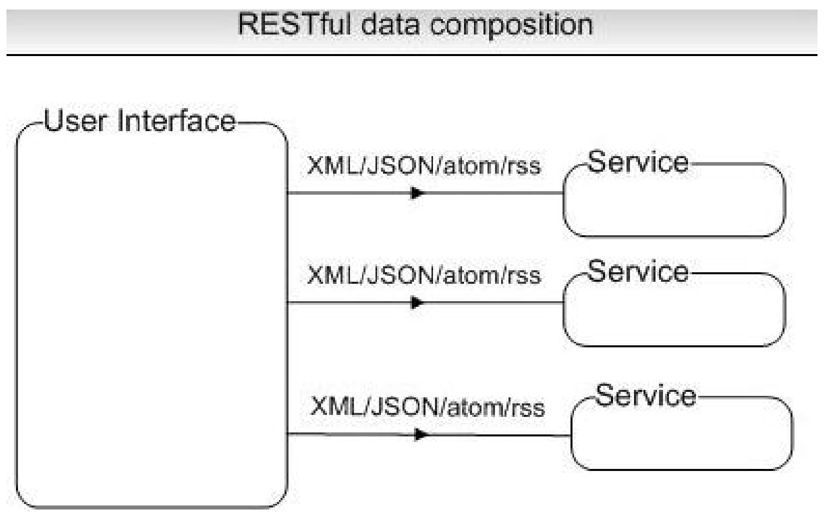
Generates the markup on the server. For example, returns generated HTML code.



Server side templating

Client side templating and data composition

Generates the markup on the client. More suited for applications that load more data from different back end service providers via Ajax. The services could return data in JSON or XML format and then add the HTML snippets for the new data via the client side templates.



Client side templating

Javascript templating is a technique to render templates on client-side (i.e. within the browser) with Javascript using a JSON data retrieved via RESTful web service calls. The template is basically HTML markup, sprayed with tags and variables that will insert values from the JSON data or execute a programming logic.

Q8. What are the pros and cons of server side versus client side templating?
A8.

Server Side Templating

Pros:

– More search engine friendly.
– Entire response message can be cached.
-Can work without JavaScript

Cons:

– Harder to mix and match different server side technologies. For example, retrieve data from both Java and Ruby based services
– Harder to send content for multiple devices. E.g. browser, mobile devices, etc.

Client Side Templating

Pros:

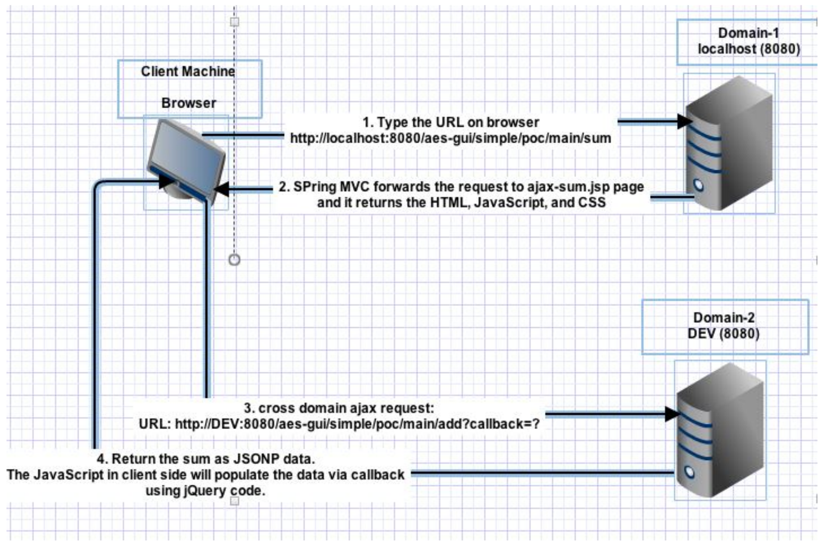
– Faster development and prototyping
– Serves JSON data to multiple devices.
– The responses are smaller as JSON data is less verbose.
– Templates and JSON data can be cached.
– Can work with multiple server side technologies like .Net, JEE, Ruby, etc

Cons:

– Less search engine friendly
– Older browsers may not work properly
– Cross browser compatibility testing is required

Q9. Can Ajax makes cross domain requests?
A9. Cross domain requests are in general prohibited by the browser. A cross domain request means, if you have a GUI application (i.e. a war) and a separate RESTful service application as a separate application running on two different domains (i.e different host-name:port no). To overcome this, you have 2 choices **JSONP** or **CORS** (Cross Origin Resource Sharing). The CORS is more industrial strength.

JSONP Example

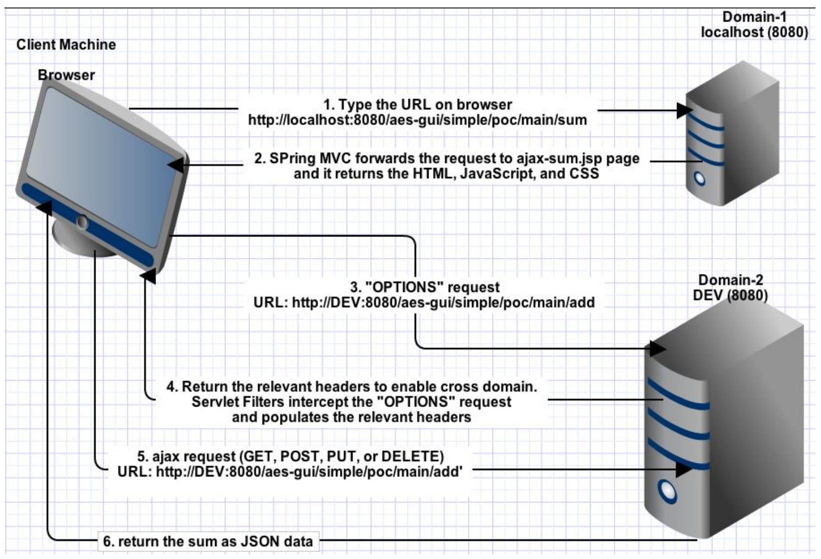

Domain-1
localhost (8080)
Client Machine
Browser
1. Type the URL on browser
http://localhost:8080/aes-gui/simple/poc/main/sum
2. SPring MVC forwards the request to ajax-sum.jsp page
and it returns the HTML, JavaScript, and CSS
Domain-2
DEV (8080)
3. cross domain ajax request:
URL: http://DEV:8080/aes-gui/simple/poc/main/add?callback=?
4. Return the sum as JSONP data.
The JavaScript in client side will populate the data via callback
using jQuery code.
Cross Domain Ajax calls

If you have a GUI application (i.e. a war) and a separate RESTful service application as a separate application running on two different domains, the you need JSONP for your Ajax to make cross domain calls. In this demo, I will be building a single war and deploy it to two different domains. For example, local and dev. The initial “sum” page will be loaded from the “local” domain, and once you click on the “add” button, the Ajax call will be made to the “dev” domain to get the calculated sum via the RESTful web service call via **jsonp callback**.

JSONP has a number of limitations like, it supports only GET requests and not PUT, POST, DELETE, etc and it does not also send headers across. **CORS** stands for Cross Origin Resource Sharing, which allows you to share GET, POST, PUT, and DELETE requests and CORS is supported by the modern browsers.The CORS make use of 2 requests.

Request 1: "**OPTIONS**" request as part of the handshake to determine if cross domain is allowed by the server.

Request 2: **GET**, **POST**, **PUT**, or **DELETE** request that performs the actual operation on the server.


Client Machine
Browser
Domain-1 localhost (8080)
1. Type the URL on browser http://localhost:8080/aes-gui/simple/poc/main/sum
2. SPring MVC forwards the request to ajax-sum.jsp page and it returns the HTML, JavaScript, and CSS
3. "OPTIONS" request URL: http://DEV:8080/aes-gui/simple/poc/main/add
Domain-2 DEV (8080)
4. Return the relevant headers to enable cross domain. Servlet Filters intercept the "OPTIONS" request and populates the relevant headers
5. ajax request (GET, POST, PUT, or DELETE) URL: http://DEV:8080/aes-gui/simple/poc/main/add'
6. return the sum as JSON data

CORS – Cross Origin Resource Sharing

Q10. How do you optimize a website's assets?
A10. CDN (Content Delivery Network) Hosting, re-organizing and refining code, minimizing round trips to/from the server, optimized caching, reducing the payload sizes and compressing payloads (e.g. JavaScript minification, combining files), add an Expires or a Cache-Control HTTP headers to cache web resources like scripts, CSS, images, etc, and moving CSS load in the head section and scripts at the bottom to load faster, etc.

Q11. Can you list some of the web development best practices?

A11.

— Favor stateless web tier. Stateless web tier means you don’t store any application data in the web server memory or file system. Keeping your web tier stateless enables you to both provide a better customer experience via better performance and your applications can scale well. If the web tier is stateless and it sits behind a load balancer, you can quickly respond to changes in application traffic by dynamically adding or removing servers. In the cloud environment you only pay for server resources that you consume.

— Use a CDN to cache static file assets and for better performance.

— Clearly separate styles via CSS and JavaScript via script files from the actual HTML content via proper includes.

— Favor using proven frameworks like angularjs, emberjs, backbone, etc for client-side MVC framewoks, Struts2, Spring MVC, etc for server side MVC, bootstrap for CSS, jQuery for manipulating the HTML DOM, etc.

— Even though the proven frameworks provide cross browser compatibility out of the box, still perform cross browser compatbility testing.

— Perform security penetration or PEN testing using tools like Skipfish from Google, Firefox plugin “tamperdata”, etc to identify security holes in your web application.

— Favor DIV element with style sheets over HTML table elements for your web page layouts.

Q12. What is a DOM, and how do you manipulate a DOM?
A12. The DOM (Document Object Model) is a programming API for documents. It defines the logical structure of documents and the way a document is accessed and manipulated. You can think of **HTML elements as objects** is DOM. It has properties, methods, and events. HTML DOM is

a standard for how to get, change, add, or delete HTML elements. For example, manipulate the DOM via JavaScript or selecting the elements via jQuery selectors. Google chrome browser's "developer Tools" or Fire fox's "Firebug" plugin can be used to view the DOM elements and perform client side debugging.

Popular Posts

♦ 11 Spring boot interview questions & answers

825 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

766 views

18 Java scenarios based interview Questions and Answers

400 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

388 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

001B: ♦ Java architecture & design concepts interview questions & answers

201 views

Bio **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 05: ♦ How a thread gets blocked or suspended in Java?

03: Servlet interview Q&A ›

Posted in FAQ JEE Job Interview Q&A Essentials**,** member-paid**,** Web basics

Tags: Java/JEE FAQs**,** JEE FAQs**,** Novice FAQs**,** TopX

Empowers you to open more doors, and fast-track

Technical Know Hows

☀️ Java generics in no time ☀️ Top 6 tips to transforming your thinking from OOP to FP ☀️ How does a HashMap internally work? What is a hashing function?

☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer