♦ 8 Java EE (aka JEE) Overview interview questions & answers

Posted on August 15, 2014 by Arulkumaran Kumaraswamipillai — No Comments ↓

Q1. When a company requires Java EE experience, what are they really asking for?

A1. Java EE (i.e. Enterprise Edition) is a collection of specifications for developing and deploying enterprise applications. In general, Java enterprise applications refer to applications written in Java & packaged as war or ear files and hosted on application servers like Tomcat, JBoss, IBM Webspehere, etc. Some of the JEE core technologies & APIs are JPA, JDBC, JNDI, EJBs, RMI, JSP, JSF, Java Servlets, XML, JMS, Java IDL, JTS, JTA, JavaMail, and JAF.

Q2. Full stack developers are in more demand. What does the term "**full stack Java developer**" mean?
A2. Have you seen job advertisements like….

"engineering team is looking for a **Full Stack Java Developer** ……"

Don't be too overwhelmed by companies looking for the following skill sets in a full-stack developer. This site covers many of these topics to increase your awareness and skills as a full stack Java developer:

Java, JEE, Spring, Hibernate, Maven, Relational databases, NoSQL databases, SQL, JavaScript, HTML, CSS, JSON, XML, Unix, UML , ERD for data modelling, AngularJS, Hadoop, AWS, Node.js, etc.

Some are mandatory, whilst the others are optional and you need to be flexible enough to learn those skills. Full stack developer is a very loose term with experience in 2 more of the categories listed below.

1) UI Frameworks such as React, AngularJS and JQuery. Javascript, HTML, and CSS skills are essential.

2) Web service development skills using Java, JEE, Spring, Hibernate, RDBMS databases & SQL.

3) Integration skills like message queues, JMS (E.g. Webspeher MQ, ActiveMQ, etc), ESB (Apache Camel, Mule), FIX Protocol, etc

4) ETL and ELT skills like Spring batch, Hadoop, Spark, Sqoop, Flume, etc.

5) Infrastructure skills like Unix, AWS, Cloud computing, System monitoring, application/system, security, etc.

The high-level architecture diagram in **Q5** shows that user interface was written in JavaScript based

⊞ Other Interview Q&A t
⊞ ▶ 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 Ir
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

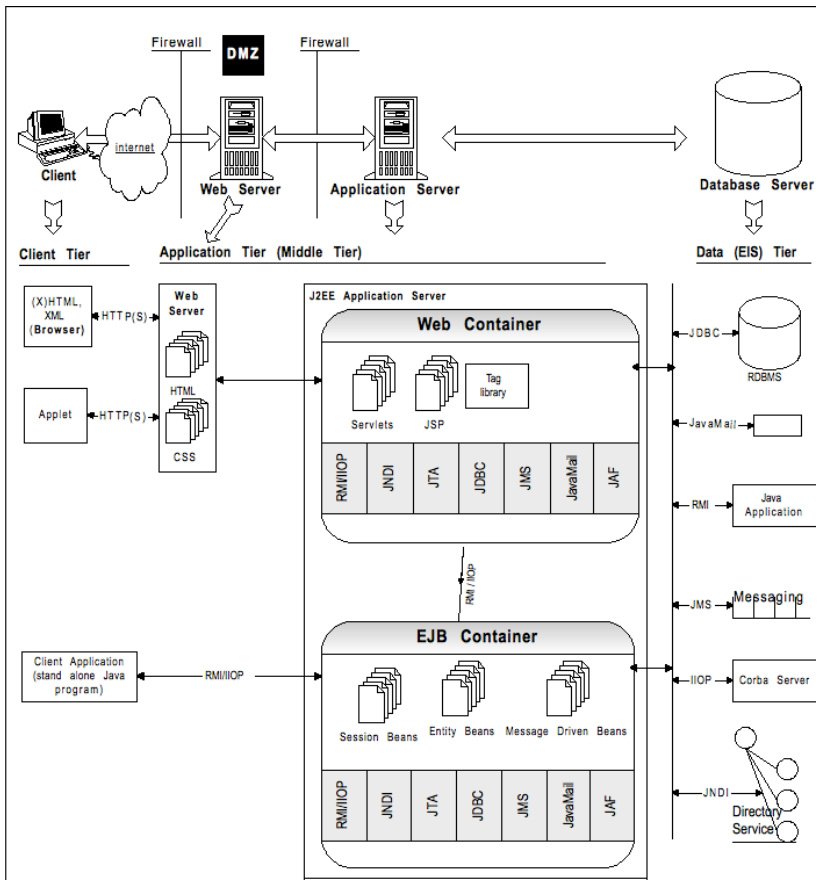
open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2
⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)
⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

technologies/frameworks, and web services were written in Java, Spring, Hibernate based technologies/frameworks. Developing this application requires a **full stack web developer**. The back-end systems can be integrated with the other systems via messaging, RESTful web services, and ETL as explained in 7+ Java integration styles & patterns interview questions & answers.

Transition from:

Core Java Developer => Full stack Web Developer => Full stack Java developer

Q3. What is Java EE? What are JEE technologies and services?
A3. Java platform Enterprise Edition or Java EE is a Java computing platform providing APIs and run time services.



JEE big picture

1) JEE APIs like

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

— javax.servlet.* for **Servlet & JSP** to service HTTP protocol.

— javax.faces.* for Java Server Faces (**JSF**) to create user interfaces.

— javax.el (Expression Language) used for binding JSF components.

— javax.enterprise.inject.* for defining injection annotations for the Contexts and Dependency Injects (**CDI**).

— javax.enterprise.context for **CDI**

— javax.ejb.* the Enterprise Java Beans (i.e. **EJB**s)

— javax.persistence.* for Java Persistence API (i.e. **JPA**)

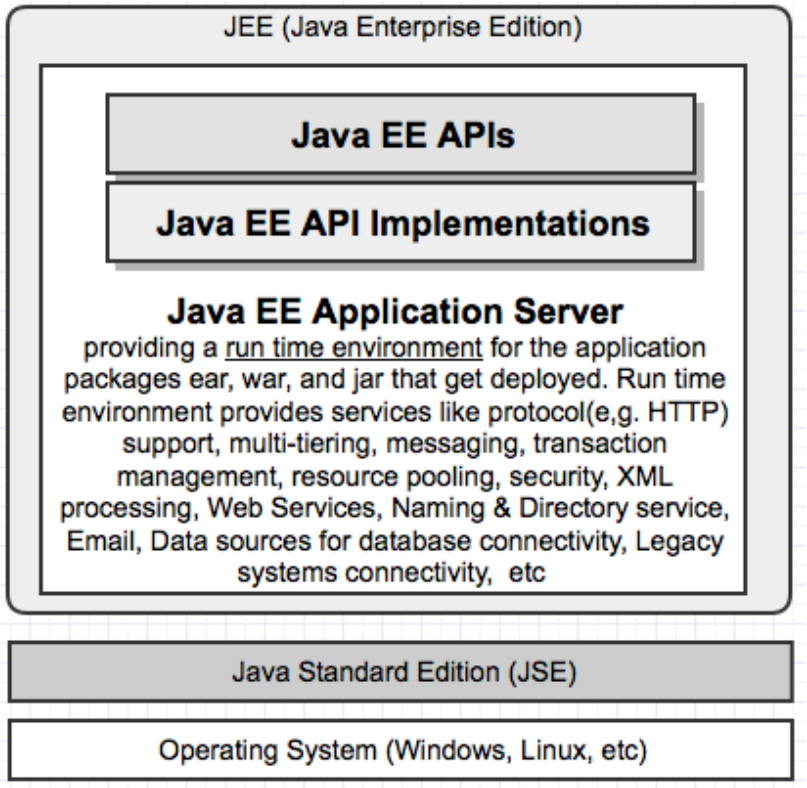
— javax.transaction.* for Java Transaction API (**JTA**)

— javax.jms.* for Java Message Service (i.e. **JMS**) to send and receive messages from Message Oriented Middlewares.

javax.resource.* for Java EE Connector Architecture (**JCA**) to integrate with EIS(Enterprise Integration Systems) systems.

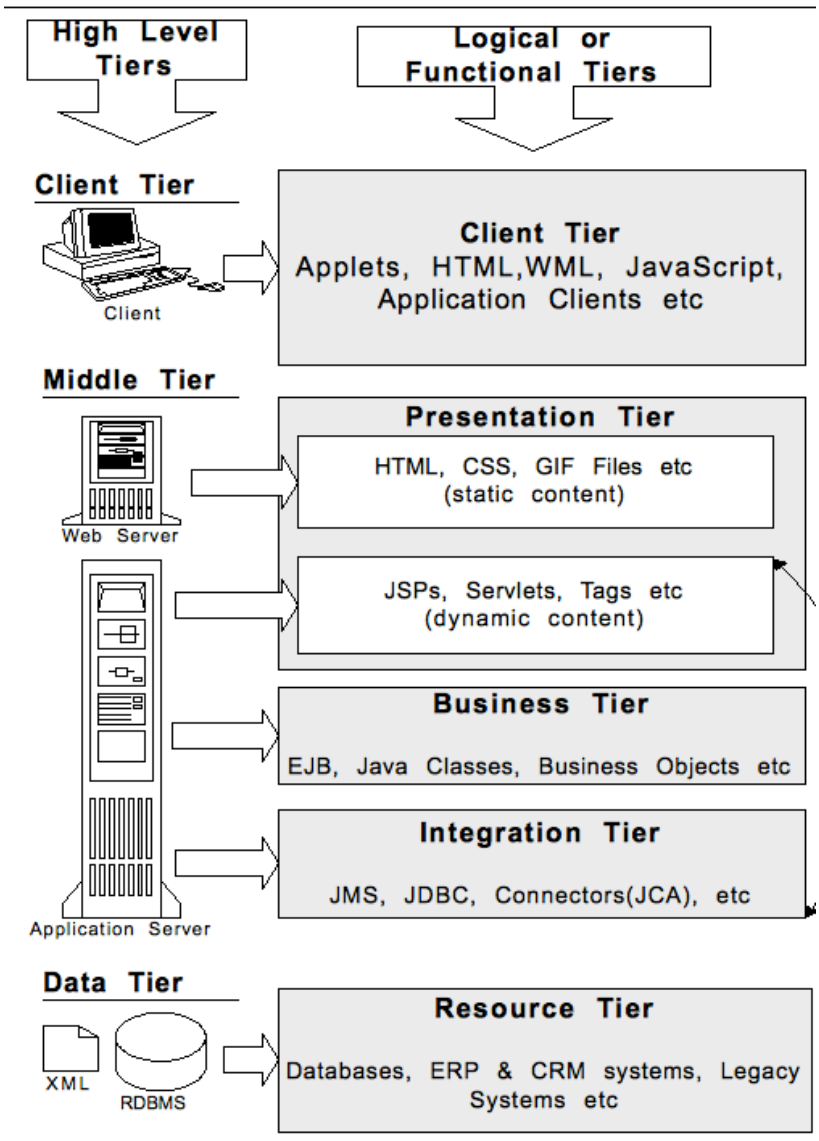
2) runtime environment

for developing and running enterprise software packages like ear (Enterprise ARchive), war (Web ARchive), or jar (Java ARchive) deployed to an JEE application server like JBoss server having web and EJB containers. The modular components running in an application server make use of annotations and conventions to configure or wire up all the components and modules. Optionally, XML based deployment descriptor files can be used to override annotations.



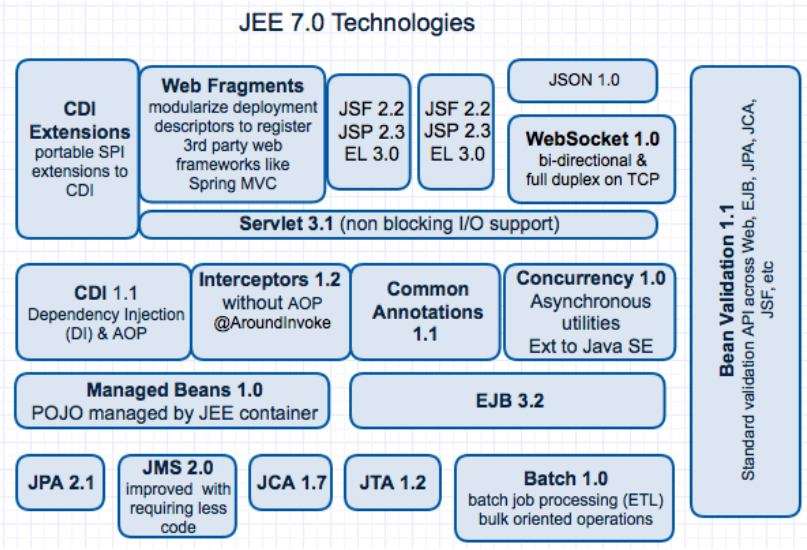
JEE Big Picture

Physically it is a 3 tier system, but logically and functionally, JEE is a multi-tiered or n-tier system.



JEE n-tier system

At the time of writing, Java EE 7 was out focusing on HTML 5 with WebSocket and JSON. WebSocket provides for two-way communication with a remote host, and Web applications can maintain bidirectional communications with server-side processes. JSON, or JavaScript Object Notation, is a lightweight data interchange format based on JavaScript and featuring a language-independent text format, which is less verbose than XML. The goal is to have a standard way to accomplish things.



JEE 7 Technology Stack

Q4. Can you give some examples of improvements in the later Java EE specification in comparison to the older JEE specs?

A4. The new specification is based on favoring “convention over configuration” for ease of development, and introduces annotations to replace the use of verbose XML for configuration.

— **CDI provides unification of DI technologies** (e.g. Guice, Spring, Seam, EJB3, and CDI itself) and introduces lifecycle to components to simplify development of new features. You can combine Seams lifecycle mappings with Spring MVC’s and with JPA’s.

— **Managed Beans was introduced to address interaction issues among the presentation layers(Servlets, JSP and JSF) and persistance layers(EJB 3.0, JTA, JCA and JPA)**. There are many form of beans in JEE like JSF Bean, EJB Bean, Spring Bean, Seam Bean, Guice CDI, Entity Beans, etc, but ultimately all are POJOs defined by the “Managed Bean 1.0” spec, and the container does provide the basic services like **Lifecycle Management** (@PostConstruct, @PreDestory), **Injection of a resource** (@Resource), and **Interception with interceptors** (@Interceptors, @ArounInvoke). Manage Bean is an SPI that was extended for this specific use.

— Introduction of **Profiles and Pruning** allow technical architects & developers to control the APIs & Specs needed for a application. Java/JEE is a huge platform and imports a lot of APIs to projects. Profiles is an introductory concept to limit the number of APIs needed for a specific application. There are pre defined profiles like Web Profile. Pruning allows removal of existing APIs or Frameworks being used by an application. JEE 6 onwards has the plug-ability to extend your Java EE with other frameworks via Extensibility points and Service Provider Interfaces (SPI).

— From Servlets 3.0 it is not required to change the descriptor file web.xml for adding the third party libraries like Struts, Spring, JSF, etc. **Servlets 3.0 introdues Web Fragments descriptor to specify the details of each library** used by the container.

— Dependency Injection simplifies EJBs.

— The persistence layer was replaced with JPA.

— Batch processing is introduced for bulk processing

— HTML 5 is embraced in JEE 7 with Web Sockets and JSON

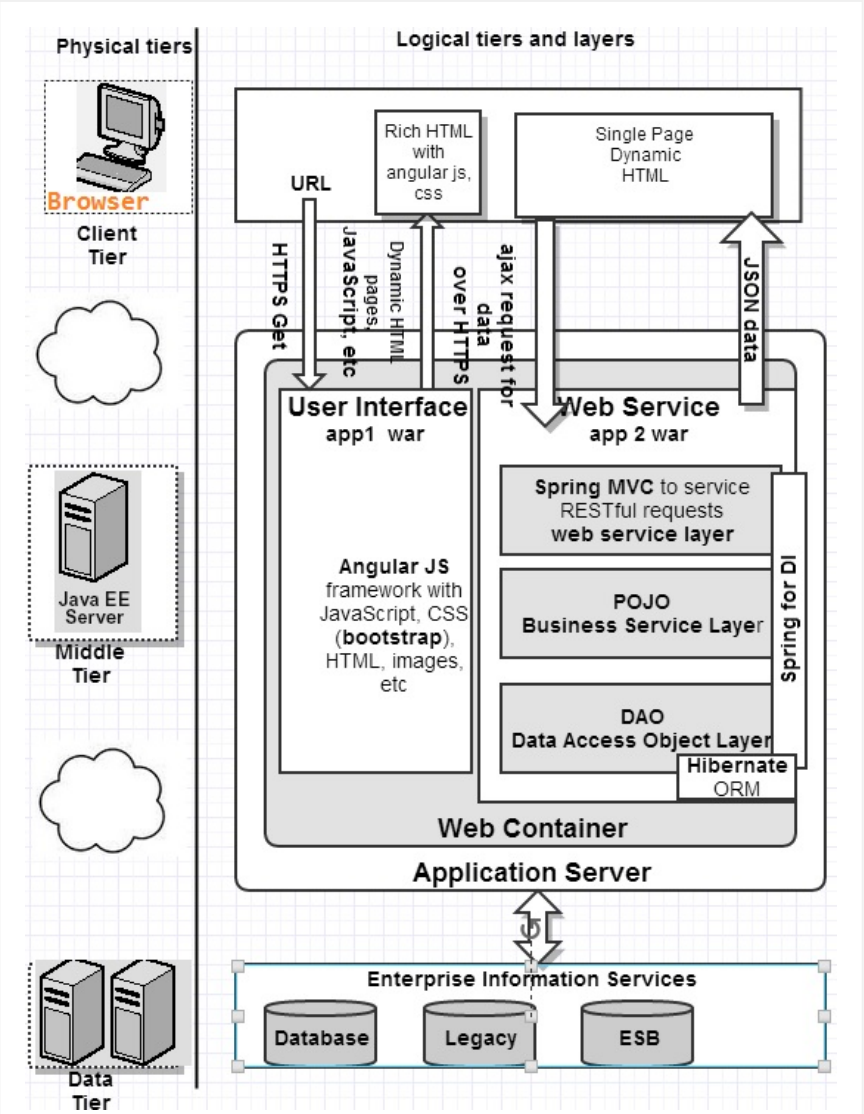
— JEE Concurrency utilities extending the Java SE Executor framework for asynchronous processing.

— Standardized validation framework across Web, EJB, JCA, etc

Spring framework had been filling some of the gaps in Java EE by focusing on POJO centric development, and has dominated the mainstream development in Java, and now JEE has embraced POJO centric development with lots of enhancements and improvements in a standard way. So, be prepared for interview questions like what is your take on Spring Vs JEE 6 or JEE 7 for future development? There are no right or wrong answers, but ask questions like

Is backward compatibility required? Does our application server support the new version of Java EE, Can I achieve everything in Java EE version 6 that can be achieved in Spring?, etc. Spring Vs. JEE will be an ongoing hot topic for a while.

Q5. Can you describe the architecture and technologies used in one of the Java EE projects you were recently involved in?
A5. This is a very popular question, and your answer will lead to other drill down questions.



Java EE high level architecture with tiers and layers

Step 1: When the url is typed like https://myhost:8080/myapp the relevant dynamic HTML files with angularjs code and all

relevant angular.js files, bootsrtap CSS files, and images are downloaded into the client browser.

Step 2: The angularjs based user interface will be a Rich Internet Application (aka RIA or Single Page Application) with tabs and rich GUI will be making ajax requests to the https://myhost:8080/my-ws to get data in JSON (JavaScript Object Notation).

The "Web Service" layer is HTTP protocol dependent, and the "Business" and "DAO" layers make us of POJOs (Plain Old Java Objects). Hence, protocol independent. This where you need to put the business logic and data access logic.

Two web archive (i.e. war) files are deployed to the same web container. One for the user interface and the other one is to retrieve data via RESTful web services. The ajax requests will only render a section of the single page. The presentation war file with HTML, CSS, and JavaScript can be hosted via CDN (i.e. Content Delivery Networks), which is a large distributed system of servers deployed in multiple data centers across the Internet. The goal of a CDN is to serve content to end-users with high availability and high performance.

Note: Like me, many have a love/hate relationship with JavaScript. Now a days, JavaScript is very popular with the rich internet applications (RIA). So, it really pays to know JavaScript. JavaScript is a client side (and server side with node.js) technology that is used to dynamically manipulate the DOM tree. There are a number of JavaScript based frameworks like **jQuery** available to make your code more cross browser compatible and simpler.What is even more popular is the JavaScript based MVW (Model View Whatever) frameworks like **angularjs**, **backbone.js**, **ember.js**, etc to build GUI.

Q6. What are the advantages of a 3-tiered or n-tiered application?
A6. 3-tier or multi-tier architectures force separation among

presentation logic, business logic and database logic. Let us look at some of the key benefits:

Manageability: Each tier can be monitored, tuned and upgraded independently and different people can have clearly defined responsibilities.

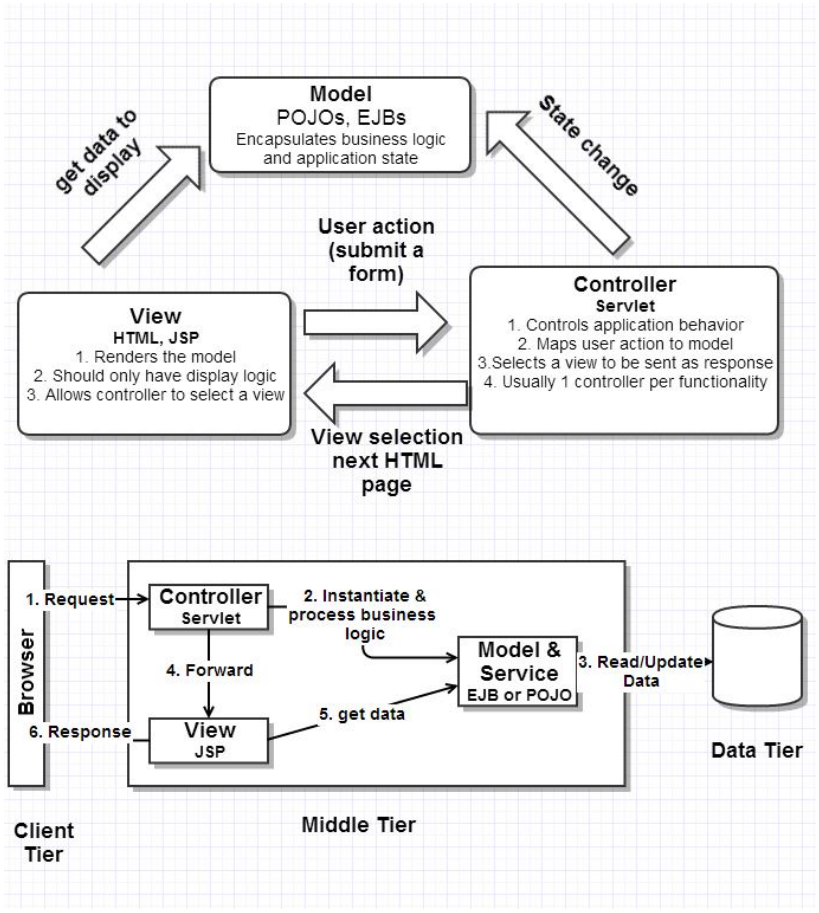
Scalability: More hardware can be added and allows clustering (i.e. horizontal scaling).

Maintainability: Changes and upgrades can be performed without affecting other components.

Availability: Clustering and load balancing can provide availability.

Extensibility: Additional features can be easily added.

Q7. Explain MVC architecture relating to JEE?
A7. This is a very popular interview question. MVC stands for Model-View-Controller architecture. It divides the functionality of displaying and maintaining of data to minimize the degree of coupling (i.e. promotes loose coupling) between components. It is often used by applications that need the ability to maintain multiple views like HTML, WML, XML based Web service, etc of the same data. Multiple views and controllers can interface with the same model. Even new types of views and controllers can interface with a model without forcing a change in the model design.



Java EE MVC pattern

The JavaScript based MVC frameworks like angularjs uses MVW (Model-View-Whatever. Where Whatever stands for "whatever works for you" — MVC – Model-View-Controller, MVP – Model-View-Presenter, MVVM – Model-View-ViewModel, etc).

Q8. What is the difference between a Web server and an application server?

A8.

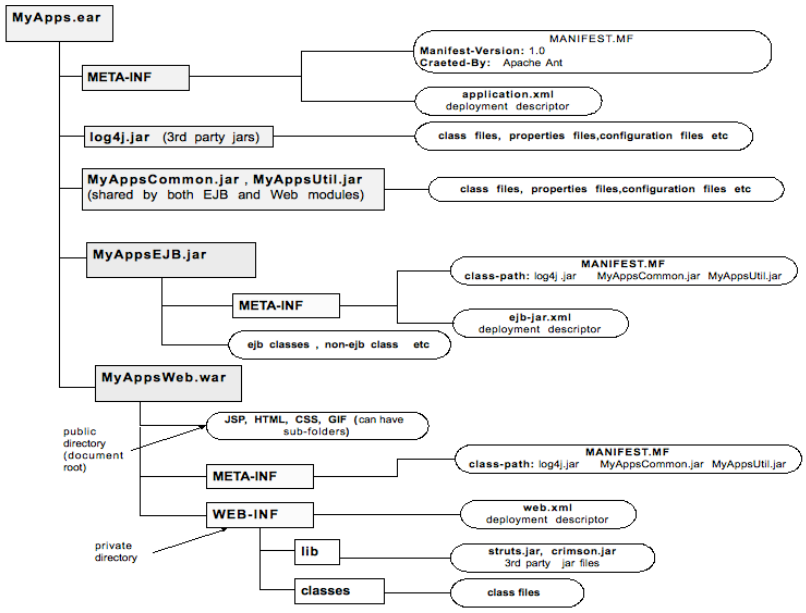
Web Server	Application Server
Supports HTTP protocol. When the Web server receives an HTTP request, it responds with an	Application Server Exposes business logic and dynamic content to the client through various protocols such

HTTP response, such as sending back an HTML page (static content) or delegates the dynamic response generation to some other program such as CGI scripts or Servlets or JSPs in the application server.	as HTTP, TCP/IP, IIOP, JRMP etc.
Uses various scalability and fault-tolerance techniques.	ses various scalability and fault-tolerance techniques. In addition provides resource pooling, component life cycle management, transaction management, messaging, security etc. Provides services for components like Web container for servlet components and EJB container for EJB components.

Q9.What are ear, war and jar files? What are JEE Deployment Descriptors?
A9.The ear, war and jar are standard application deployment archive files. Since they are a standard, any application server (at least in theory) will know how to unpack and deploy them.

An EAR file is a standard JAR file with an ".ear" extension, named from Enterprise ARchive file. A JEE application with all of its modules is delivered in EAR file. JAR files can't have other JAR files. But EAR and WAR (Web ARchive) files can have JAR files.

An EAR file contains all the JARs and WARs belonging to an application. JAR files contain the EJB classes and WAR files contain the Web components (JSPs, Servlets and static content like HTML, CSS, GIF etc). The JEE application client's class files are also stored in a JAR file. EARs, JARs, and WARs all contain one or more XML-based deployment descriptor(s).



JEE ear, war, and jar

In EJB 3.1 the **EJB module can be packaged within the web module** in two different ways.

1) Package an ejb module into a jar, and place it under WEB-INF/lib folder.
2) Compile the ejb classes into WEB-INF/classes and put the deployment descriptor in META-INF folder in the web module.

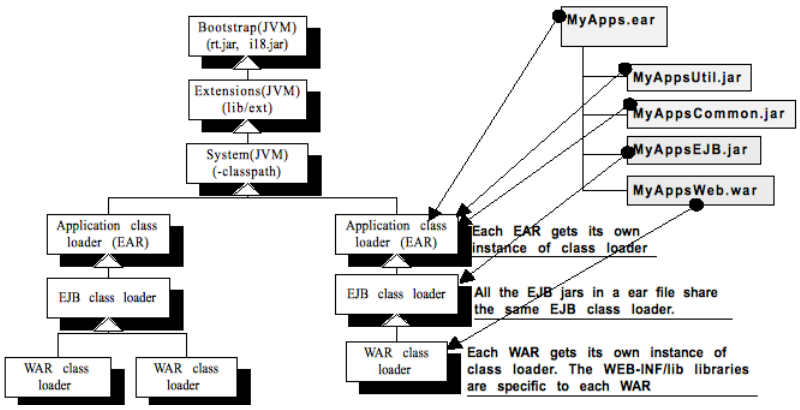
A **deployment descriptor** is an XML based text file with an ".xml" extension that describes a component's deployment settings. A JEE application and each of its modules has its own deployment descriptor.

application.xml: is a standard JEE deployment descriptor for ear files, and includes structural information: EJB jar modules, Web war modules, etc.

ejb-jar.xml: is a standard deployment descriptor for an EJB module.

web.xml: is a standard deployment descriptor for a Web module.

Q10.Can you explain the JEE class loader hierachy?
A10.JEE application server sample class loader hierarchy is shown below. As per the diagram the JEE application specific class loaders are children of the “System –classpath” class loader. When the parent class loader is above the “System –classpath” class loader in the hierarchy as shown in the diagram (i.e. bootstrap class loader or extensions class loader) then child class loaders implicitly have visibility to the classes loaded by its parents. When a parent class loader is below a “System -classpath” class loader in the hierarchy then the child class loaders will only have visibility into the classes loaded by its parents only if they are explicitly specified in a manifest file (MANIFEST.MF) of the child class loader.



JEE sample class loader hierarchy. Actual vendor server class loaders can vary.

Example As per the diagram, if the EJB module MyAppsEJB.jar wants to refer to MyAppsCommon.jar and MyAppsUtil.jar we need to add the following entry in the MyAppsEJB.jar’s manifest file MANIFEST.MF.

class-path: MyAppsCommon.jar MyAppsUtil.jar

This is because the application (EAR) class loader loads the MyAppsCommon.jar and MyAppsUtil.jar. The EJB class loader loads the MyAppsEJB.jar, which is the child class loader of the application class loader. The WAR class loader loads the MyAppsWeb.war.

Every JEE application or EAR gets its own instance of the application class loader. This class loader is also responsible for loading all the dependency jar files, which are shared by both Web and EJB modules. For example third party libraries like log4j, utility (e.g. MyAppsUtility.jar) and common (e.g. MyAppsCommon.jar) jars etc. Any application specific exception like MyApplicationException thrown by an EJB module should be caught by a Web module. So the exception class MyApplicationException is shared by both Web and EJB modules.

The key difference between the EJB and WAR class loader is that all the EJB jars in the application share the same EJB class loader whereas WAR files get their own class loader. This is because the EJBs have inherent relationship between one another (i.e. EJB-EJB communication between EJBs in different applications but hosted on the same JVM) but the Web modules do not. Every WAR file should be able to have its own WEB-INF/lib third party libraries and need to be able to load its own version of converted logon.jsp servlet. So each Web module is isolated in its own class loader.

So if two different Web modules want to use two different versions of the same EJB then we need to have two different ear files. Class loaders use a delegation model where the child class loaders delegate the loading up the hierarchy to their parent before trying to load it itself only if the parent can’t load it. But with regards to WAR class loaders, some application servers provide a setting to turn this behavior off (DelegationMode=false).

As a general rule classes should not be deployed higher in the hierarchy than they are supposed to exist. This is because if you move one class up the hierarchy then you will have to move other classes up the hierarchy as well. This is

because classes loaded by the parent class loader can't see the classes loaded by its child class loaders (uni-directional bottom-up visibility).

You may also like

1. Top 50+ FAQ Java EE Interview Questions & Answers

2. 30+Java Web Basics Interview Questions and Answers

3. 50+ Java Web Services Interview Questions & Answers

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**

‹ ♦ Java Garbage Collection interview Q&A to ascertain your depth of Java knowledge

Java EE interview questions and answers ›

Posted in FAQ JEE Job Interview Q&A Essentials, JEE Overview, member-paid

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

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

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

 ↑