Q) What is webservice ?

=) A web service can be defined as:

* The method of communication between two devices over the network.

* It is a collection of standards or protocols for exchanging information between two devices or applications.

* Web services are language independent ways of communication.

Examples : Weathers apps
location apps (Maps)

Q) What are JAX-WS and JAX-RS ?

=) JAX-WS (Java API for XML Web Services) is a Java API for creating & consuming SOAP web services.

* JAX-WS uses XML for message formating & is more complex, slow due to usage of WSDL (Web services Description language).

=) JAX-RS (Java API for Restful Web services) is a Java API for creating RESTful web services.

* JAX-RS uses HTTP methods for communication and is based on stateless, client-server, cacheable

Note

* JAX-WS is suitable for enterprise-level applications that require more security & reliable messaging.

* JAX-RS is suitable for web applications, mobile applications & microservices.

Q) JAX-WS Concepts ?

=) JAX-WS uses the following annotations to simplify the development of web services:

* @WebService : Defines a class as a webservice end point.

* @WebMethod : Exposes a method as a web service operation

@WebParam: Defines parameters for web service methods

@WebResult: Defines the return value of a web service method

=> JAX-WS uses WSDL that is an XML-based language which provides a model for describing web services.

=> JAX-WS follows TOP-Down & Bottom-up Approaches.

=> TOP-Down (contract-first) starts with a WSDL file & generate Java classes whereas Bottom-up is vice-versa.

Q) Ways of implementing JAX-WS ?

=> RPC (Remote Procedure call) & Document-oriented are way of implementing JAX-WS.

=> RPC focuses on method calls & parameters & is tightly coupled.

=> Document-oriented focuses on exchanging entire XML documents making it more flexible & loosely coupled.

Q) JAX-RS concepts ?

=> JAX-RS uses annotations to simplify the development & deployment of web services.

@Path: Specifies the relative path (route) for a resource, class or method.

@GET, @POST, @PUT, @DELETE: Specifies the HTTP request type.

@PRODUCES: Specifies the response media types. Similar to @GET but we need to specify MEDIATYPE

@CONSUMES: Specifies the accepted request Media Types, Similar to @POST but MEDIATYPE is required.

=> Resources are typically POJOs that are annoted to expose them as web resources.

Q) Ways to implement JAX-RS ?

=> Jersey, a reference implementation of JAX-RS, known for its extensive features and ease of use.

=> RESTEasy, a robust implementation by JBoss, offering tight integration with JBoss application server & a powerful client framework.

Q) REST & Constraints of REST ?

=) REST is a software architectural style that defines the set of rules to be used for creating web services.

=) Interaction in REST is based on HTTP requests.

=) A REST system consists of a client who requests for the resources and a server who has the resources.

Architectural Constraints

* Uniform Interface:
It suggests that there should be a uniform way of interacting with the server irrespective of device or type of application.
→ Resource Based
→ Manipulation of resources through representations.
→ Self-descriptive message.
→ Hypermedia as the Engine of Application State (HATEOAS) states that the client should access other resources easily

* Stateless * Cacheable
* Client server * Code on demand
* Layered system

Q) SOAP vs REST ?

=) SOAP vs REST

* SOAP is a protocol. REST is an architectural style.

* SOAP can't use REST, REST can use SOAP.

* SOAP uses services interfaces to expose the business logic, REST uses URI to expose business logic.

* JAX-WS is the Java API for SOAP web services, JAX-RS is the Java API for RestFul web services.

* SOAP permits XML format data only, REST permits text, HTML, JSON, etc.

* SOAP requires more bandwidth as compared to REST.

* SOAP defines much strict standards to be followed.