Popular Web Services Interview Questions and Answers

Q. What are web services?

They are client, server or provider, consumer applications that communicate on the network using the http protocol and exchange messages or data in xml or various other formats like JSON.

Q. What is the difference between SOA and a Web service?

A. SOA is a design and architecture to implement other services. SOA can be easily implemented using various protocols such as HTTP, HTTPS, JMS, SMTP, RMI, IIOP, RPC etc.

Web service is an implementation technology and one of the ways to implement SOA. You can build SOA based applications without using Web services – for example by using other traditional technologies like Java RMI, EJB, JMS based messaging, etc. But what Web services offer is the standards based and platform-independent service via HTTP, XML, SOAP, WSDL and UDDI, thus allowing interoperability between heterogeneous technologies such as J2EE and .NET.

Q. Why not favor traditional style middle-ware such as RPC, CORBA, RMI and DCOM as opposed to Web services?

A. The traditional middle-wares tightly couple connections to the applications and it can break if you make any modification to your application. Tightly coupled applications are hard to maintain and less reusable. Generally do not support heterogeneity. Do not work across Internet.

Web Services support loosely coupled connections. The interface of the Web service provides a layer of abstraction between the client and the server. The loosely coupled applications reduce the cost of maintenance and increases re-usability. Web services are language and platform independent. You can develop a Web service using any language and deploy it on to any platform, from small device to the largest supercomputer. Web service uses language neutral protocols such as HTTP and communicates between disparate applications by passing XML messages to each other via a Web API. Do work across internet, less expensive and easier to use.

Q. What is meant by JAX-WS and JAX-RS?

JAX-WS stands for java api for xml based web services and has the specification and api for soap based web services.

JAX-RS stands for java API for xml based Rest Services and has the specification and api to implement RESTFul web services.

SOAP

Q. Also explain how to call this WS after all the stub/skeletons are created.

Three Steps:

1. Create the instance of the Service Class by passing the WSDL URL to the constructor.
2. Create the porttype by invoking the get method on the Service created in step one
3. Call the web service methods by passing in the appropriate parameters.

Q. How to consume a web service given a WSDL. Explain all the tools required to create the stubs/skeleton, configuration etc.

A. Three Steps:

1. Generate the stubs from the wsdl using wsdl2java
2. Create an instance of the porttype by giving it the wsdl url
3. Invoke the method on the object.

Q. What are the steps to implement user name token profile?

A. Two Steps:

1. Implement the call back handler class that can return the password for the given user id.
2. Configure the User Name token profile security in the cxf-servlet.xml

Q. What are the steps to configure Basic Authentication?

A. Two Steps:

1) Configure the user in tomcat-users.xml or other application servers configuration
2) Configure the basic authentication using the login-config and auth-constratints elements in the web.xml of our JEE Web Application.

Q. What are the steps to implement java first web service?

A. Four Steps:

1. Create the Web Services Interface , Implementation and other bean Classes
2. Annotate the web service interface and implementation with the

3. Annotate the beans with JAXB annotations
4. Define the Endpoint in the cxf-servlet.xml (When CXF Is Used)

Q. List the steps to exchange files using MTOM?

A. Two Steps:

1. Use the DataHandler type as the bean property or parameters to the web service method.
2. Enable the MTOM property on the endpoint in the cxf-servlet.xml

Q. Explain the structure of a WSDL File?

Abstract Section:

Wsdl:types – Has the xml schema that can be used to build the messages

Wsdl:messages: the request and response xml messages built using the schema in the types section above.

Wsdl:Operation: Each operation that the web service exposes.

Wsdl:PortType: Groups the operations or web service methods together.

Physical Portion:

Wsdl:Binding – We define the binding for all the operations which controls how the soap message is generated.

Wsdl:service: The location/url of the service using which it can be accessed.

Q. Which web services stacks are familiar with?

A. Apache CXF, Apache AXIS, Metro etc

Q. What is a SOAP Engine?

A. SOAP Engine does two things:

Dispatches the SOAP Requests to the appropriate endpoints

Serializes the soap messages to language objects and De-Serializes the language objects to soap messages.

Q. What WS Standards are you aware off?

A.

WS-Security

MTOM

WS-Addressing

WS-Policy

WS-Reliable Messaging

Q. What are the different approaches to developing SOAP based Web service?

A. There are 2 approaches to develop SOAP based Web Services..

The contract-first approach, where you define the contract first with XSD and WSDL and the generate the Java classes from the contract.

The contract-last or code first approach where we define the Java classes first and then generate the contract, which is the WSDL file from the Java classes.

Q. What are the pros and cons of each approach, and which approach would you prefer?

A. **Contract-first Web service**

PROS:

Clients are decoupled from the server, hence the implementation logic can be revised on the server without affecting the clients.

Developers can work simultaneously on client and server side based on the contract both agreed on.

You have full control over how the request and response messages are constructed -- for example, should "status" go as an element or as an attribute? The contract clearly defines it. You can change OXM (i.e. Object to XML Mapping) libraries without having to worry if the "status" would be generated as "attribute" instead of an element. Potentially, even Web service frameworks and tool kits can be changed as well from say Apache Axis to Apache CXF, etc

CONS:

More upfront work is involved in setting up the XSDs and WSDLs. There are tools like XML Spy, Oxygen XML, etc to make things easier. The object models need to be written as well.

Developers need to learn XSDs and WSDLs in addition to just knowing Java.

Contract-last Web service

PROS:

Developers don't have to learn anything related to XSDs, WSDLs, and SOAP. The services are created quickly by exposing the existing service logic with frameworks/tool sets. For example, via IDE based wizards, etc.

The learning curve and development time can be smaller compared to the Contract-first Web service.

CONS:

The development time can be shorter to initially develop it, but what about the on going maintenance and extension time if the contract changes or new elements need to be added? In this approach, since the clients and servers are more tightly coupled, the future changes may break the client contract and affect all clients or require the services to be properly versioned and managed.

In this approach, The XML payloads cannot be controlled. This means changing your OXM libraries could cause something that used to be an element to become an attribute with the change of the OXM.

So, which approach will you choose?

The best practice is to use "contract-first", and here is the link that explains this much better with examples --> contract-first versus contract-last web services In a nutshell, the contract-last is more fragile than the "contract-first". You will have to decide what is most appropriate based on your requirements, tool sets you use, etc.

How did you secure your SOAP Based Web Services?

A.SOAP WS has better support for security

What tools do you use to test your Web Services?

A. SoapUI tool for SOAP WS and Chrome Advanced REST Client for RESTFul services.

What types of operations are available in WSDL?

A. There are four operations available:

1. One-way, where the operation can receive a message but will not return a response.

2. Request-response, where the operation can receive a request and will return a response.

3. Solicit-response, where the operation can send a request and will wait for a response.

4. Notification, where the operation can send a message but will not wait for a response.

RESTFul

Q. What are the steps to design a REST Web Service?

A. Four Steps:

Identify the objects/resources using the class diagram. There are the nouns in the object model.

Create the URIs to access these resources.

Choose the HTTP methods to bind to the operations on the resources.

Identify the data format that needs to be supported and come up with sample requests and responses.

Q. What are the steps to create a REST web service?

A. Four Steps:

Create the resource interface and implementation classes.

Bind the methods on the resources to HTTP Methods using the JAX-WS (@GET, @POST, @PUT, @DELETE ETC) annotations.

Bind the resources and their methods to URI Path using the JAX-WS (@Path) annotations.

Configure/Publish the Resource in the cxf-servlet.xml

Q. What is the difference between HTTP POST and PUT requests?

The HTTP methods POST and PUT aren't the HTTP equivalent of the CRUD's create and update. They both serve a different purpose. It's quite possible, valid and even preferred in some occasions, to use POST to create resources, or use PUT to update resources.

Use PUT when you can update a resource completely through a specific resource.

If you do not know the actual resource location, for instance, when you add a new object, but do not have any idea where to store it, you can POST it to an URL, and let the server decide the actual URL.

As soon as you know the new resource location, you can use PUT again to do updates to the blue stapler article. But as said before: you CAN add new resources through PUT as well. The next example is perfectly valid if your API provides this functionality

PUT and POST are both unsafe methods. However, PUT is idempotent, while POST is not

Q. What kind of output formats can one generate using RESTful web service?

Xml,JSON,TEXT ETC

Q. What all tools have you used to write RESTful web service?

Eclipse IDE,Apache CXF, Chrome REST plugin for testing

5) How is JAXB related to RESTful web services?

JAXB can be used to bind the java objects to XML and JSON that gets generated from our rest services.

Q. What will you do when an error code has to be returned to the client

OR

How will you handle application error scenarios in RESTful web service

A. There are two types of errors.

1) Standard Errors

2) Application Errors

Standard Errors Can be handled in three ways:

We can returns the error code using the JAX-RS Response Objects Methods

We can throw a WebApplicationException and set the http error status on it.

We can throw a specific exception.

Application Errors or exceptions can be handled using one of the three ways above or my writing JAX-RS ExceptionMappers.There exception mappers should be marked with the @Provder JAX-RS annotations and should be configured in the cxf-servlet.xml.

Q. Can a RESTful web service generate output in various formats based on some parameter received from the client?

A. Yes. Based on the accept http header the RESTFul provider will know what MIME TYPE data the client can accept send out the appropriate response back.

Q. With a RESTful web service, whose state is getting transferred and how?

It is the state of the application that is being transferred between the provider and the consumer.

Example:

A shopping cart with items in it

Airline Ticket Reservation Information of a passenger between a KIOSK Check Application (Client) and Rest web service application (Provider).

Q. Which take up more network bandwidth SOAP or REST?

SOAP is more verbose. Because of the SOAP XML Elements (SOAP – Envelope ,Header, Body) overhead SOAP needs more bandwidth.

Q. SOAP WS or REST?

A. In general, a REST based Web service is preferred due to its simplicity, performance, scalability, and support for multiple data formats. SOAP is favored where service requires comprehensive support for security and transactional reliability.

The answer really depends on the functional and non-functional requirements. Does the service expose data or business logic? (REST is a better choice for exposing data, SOAP WS might be a better choice for logic).