**Soap Basic Concepts**

Web services are consumer provider application which interacts between two using SOAP,JSON etc and use HTTP methods for interaction.

**Benefits :**

* Interoperability – Any web service on any environment is able to interact among one another.
* Loosely Coupled – No hardly coupled.
* Extensibility – New features can be added time to time.

**Types:**

1. SOAP
   1. Type – XML
   2. Methods – HTTP Post Methods
2. Restful Web Services
   1. Type – Multiple Data Formats
   2. Methods – Http methods

**XML :**

Extensible Markup language. HTML allows us to use predefined tags only but it allows us to create our own markup tags. For instance, if a student is used here, id and name,

<Student>

<Id>1234</Id>

<Name>Nikhil Prabhat</Name>

</Student>

It has data as well as meta-data. Data refers what value needs to be passed and meta-data refers what needs to be done with the data.

XML can be used at three places.

1. Data Exchange – SOAP and REST web services are using XML
2. Configuration – like web.xml, pom.xml etc
3. Save, Manipulate and Present - like saving data as xml and getting it converted to HTML

**XSD :**

XML Schema Definition. It is basically the blueprint or grammar of the xml.It is also a xml file with .xsd extension. If an XML follows XSD, it is considered to be valid XML.It defines elements, names, attributes, namespaces, order, number of occurrences,restrictions etc.

It works as a contract between two apps sharing data with xml in between. Like we write configuration file for db connections, we are provided with xsd, and according to that, we write our configurations.

**NameSpaces :**It is basically used to uniquely identify the XML components. In java world, it is considered as merely packages.

Targetnamespace : [www.amazon.com/order](http://www.amazon.com/order), [www.ebay.com/order](http://www.ebay.com/order)

Prefix is used so that we don’t need to call the entire URL with the help of xml name space.

<xmlns: amz = “www.amazon.com/oder”>

<xmlns: ebay = “www.ebay.com/order” >

<order xmlns: amz=”www.amazon.com/order”>

<amz:lineitem/>

<amx:ordered/>

</order>

**Element** : It is building block of XML element.It is used as a container to store text elements, attributes and media objects.

**ComplexType** :

Complex Element is an XML element which can contain other elements and/or attributes. We can create a complex element in two ways −

* Define a complex type and then create an element using the **type** attribute
* Define a complex type directly by naming

Sequence signifies the sequence in which object will come.

<xs:complexType name = "StudentType">

<xs:sequence>

<xs:element name = "firstname" type = "xs:string"/>

<xs:element name = "lastname" type = "xs:string"/>

<xs:element name = "nickname" type = "xs:string"/>

<xs:element name = "marks" type = "xs:positiveInteger"/>

</xs:sequence>

<xs:attribute name = 'rollno' type = 'xs:positiveInteger'/>

</xs:complexType>

<xs:element name = 'student' type = 'StudentType' />

**SimpleType :**

The simpleType element defines a simple type and specifies the constraints and information about the values of attributes or text-only elements.

<xs:element name="age" type = “tns:ageSimpletype”>  
  <xs:simpleType name = “ageSimpleType”>  
    <xs:restriction base="xs:integer">  
      <xs:minInclusive value="0"/>  
      <xs:maxInclusive value="100"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>

**Choice :**

It lets the xml to include one element only

**All :**

For all , all elements should come but the order doesn’t matter.

To control number of elements in xml, minOccurs or maxOccurs are used.

**ElementFormDefault** : It is either qualified or unqualified. Qualified signifies it should have namespace and unqualified will compain if it has a namespace. This is just for the XML parser.

**Attributes :**

Attributes are almost similar to elements. There is no rule when to use attributes or when to elements. Both provide same information.

<person>  
  <sex>female</sex>  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>

<person sex="female">  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>

In the first case, sex is element and in the second case, sex is attribute.

**Serialisation and Deserialisation :**

Serialisation is the process of converting java objects into byte stream and deserialization is the process of converting byte streams into java objects.

For serializing the object, we call the **writeObject()** method of ObjectOutputStream class, and for deserialization we call the **readObject()** method of ObjectInputStream class.

In web browsers, generally the parsers are already there.

SOAP web services should be used shen formal contract is required, non-functional requirements like security and transactions management, reliable asynchronous processing.

WSDL – Web Service Description Language (Contract in SOAP)

WSDL signifies what the web service provides as well as how the web service provides.

Abstract/What :

Definitions – types, messages, operation , porttype

Physical/How :

Binding, Service

There are multiple types of soap binding styles. It affects basically three things:

* Soap payload
* Validation
* Operation Name Soap Message

**Soap Web Service Design :**

1. Top-Down or WSDL first
   1. Create the WSDL file
   2. Generate the java stubs using tools like wsdl2java
   3. Implement the web services endpoint
2. Bottom-Up or Code first
   1. Write Java code and annotate
   2. Generate the WSDL from the code using java2wsdl

**JAX-WS (Java API for XML Based Web Service)**

It consists of two things

1. Specification
   1. It refers to engine like CXF, GlassFish
2. API
   1. Annotations
   2. Developers

Developers use those annotations and are comprehended by engine

**Few Important Annotation :**

JAX-WS Annotation

@WebService – It refers to the endpoint which is responsible for the interaction of web service

@WebMethod - It is used to mark business method that are exposed to web service clients.Be default, all public methods in class are exposed

@WebResult – It is used to customize the name of the XML element that represents the return value in case if we don’t want to use the class name.

@WebFault – It is used on exception class so that it can be converted to soap fault

@SoapBinding – It allows to bind the web service in the desired Binding style

JAXB Annotation

@XMLAccessorType – All the JAXB annotations will be used at method or field level if XMLAccessType.FIELD

@XMLType – It is used for proper ordering of elements and the class will have its own complex type. If we don’t give a name, be default , class name will be used.

@XMLRootElement – signifies the root element

**JAXB**

It stands for Java Architecture for XML Binding. As hibernate for SQL, same JAXB for xml

Java JAXB XML

**Marshalling and Unmarshalling:**

Marshalling is the process of converting java objects into XML. Umarshalling is the process of converting XML into java objects.

**Understanding the WSDL:**

* Porttype contains operation which jots down all the methods exposed by Web Service.
* When we have more than two parameters for operations, then we use params as message
* Binding is just about how the service accepts request and returns response.
* Service has port which follows or connects to binding, which helps us to know where the service is deployed

