**RESTful** Web Services are REST architecture based web services. In REST Architecture everything is a resource. RESTful web services are light weight, highly scalable and maintainable and are very commonly used to create APIs for web based applications.

What is REST ?

REST stands for **RE**presentational **S**tate **T**ransfer. REST is web standards based architecture and uses HTTP Protocol for data communication. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods. REST was first introduced by Roy Fielding in 2000.

In REST architecture, a REST Server simply provides access to resources and REST client accesses and presents the resources. Here each resource is identified by URIs/ global IDs. REST uses various representations to represent a resource like text, JSON and XML. Now a days JSON is the most popular format being used in web services.

HTTP Methods

Following well known HTTP methods are commonly used in REST based architecture.

* **GET** - Provides a read only access to a resource.
* **PUT** - Used to create a new resource.
* **DELETE** - Used to remove a resource.
* **POST** - Used to update a existing resource or create a new resource.
* **OPTIONS** - Used to get the supported operations on a resource.

RESTFul Web Services

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

Web services based on REST Architecture are known as RESTful web services. These web services use HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.

Creating RESTFul Web Service

This tutorial will create a web service say user management with following functionalities:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **HTTP Method** | **URI** | **Operation** | **Operation Type** |
| 1 | **GET** | /UserService/users | Get list of users | Read Only |
| 2 | **GET** | /UserService/users/1 | Get User with Id 1 | Read Only |
| 3 | **PUT** | /UserService/users/2 | Insert User with Id 2 | Idempotent |
| 4 | **POST** | /UserService/users/2 | Update User with Id 2 | N/A |
| 5 | **DELETE** | /UserService/users/1 | Delete User with Id 1 | Idempotent |
| 6 | **OPTIONS** | /UserService/users | List the supported operations in web service | Read Only |

<http://www.tutorialspoint.com/restful/restful_environment.htm>

* SOAP stands for **S**imple **O**bject **A**ccess **P**rotocol
* SOAP is an application communication protocol
* SOAP is a format for sending and receiving messages
* SOAP is platform independent
* SOAP is based on XML
* SOAP is a W3C recommendation

## Why SOAP?

It is important for web applications to be able to communicate over the Internet.

The best way to communicate between applications is over HTTP, because HTTP is supported by all Internet browsers and servers. SOAP was created to accomplish this.

SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

## SOAP Building Blocks

A SOAP message is an ordinary XML document containing the following elements:

* An Envelope element that identifies the XML document as a SOAP message
* A Header element that contains header information
* A Body element that contains call and response information
* A Fault element containing errors and status information

All the elements above are declared in the default namespace for the SOAP envelope:

<http://www.w3.org/2003/05/soap-envelope/>

and the default namespace for SOAP encoding and data types is:

<http://www.w3.org/2003/05/soap-encoding>

## Syntax Rules

Here are some important syntax rules:

* A SOAP message MUST be encoded using XML
* A SOAP message MUST use the SOAP Envelope namespace
* A SOAP message MUST use the SOAP Encoding namespace
* A SOAP message must NOT contain a DTD reference
* A SOAP message must NOT contain XML Processing Instructions

## Skeleton SOAP Message

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Header>  
...  
</soap:Header>  
  
<soap:Body>  
...  
  <soap:Fault>  
  ...  
  </soap:Fault>  
</soap:Body>  
  
</soap:Envelope>

## The SOAP Envelope Element

The required SOAP Envelope element is the root element of a SOAP message. This element defines the XML document as a SOAP message.

### Example

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  ...  
  Message information goes here  
  ...  
</soap:Envelope>

## The xmlns:soap Namespace

Notice the xmlns:soap namespace in the example above. It should always have the value of: "http://www.w3.org/2003/05/soap-envelope/".

The namespace defines the Envelope as a SOAP Envelope.

If a different namespace is used, the application generates an error and discards the message.

## The encodingStyle Attribute

The encodingStyle attribute is used to define the data types used in the document. This attribute may appear on any SOAP element, and applies to the element's contents and all child elements.

A SOAP message has no default encoding.

#### Syntax

soap:encodingStyle="*URI*"

#### Example

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  ...  
  Message information goes here  
  ...  
</soap:Envelope>

## The SOAP Header Element

The optional SOAP Header element contains application-specific information (like authentication, payment, etc) about the SOAP message.

If the Header element is present, it must be the first child element of the Envelope element.

**Note:** All immediate child elements of the Header element must be namespace-qualified.

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Header>  
  <m:Trans xmlns:m="http://www.w3schools.com/transaction/"  
  soap:mustUnderstand="1">234  
  </m:Trans>  
</soap:Header>  
...  
...  
</soap:Envelope>

The example above contains a header with a "Trans" element, a "mustUnderstand" attribute with a value of 1, and a value of 234.

SOAP defines three attributes in the default namespace. These attributes are: mustUnderstand, actor, and encodingStyle.

The attributes defined in the SOAP Header defines how a recipient should process the SOAP message.

## The mustUnderstand Attribute

The SOAP mustUnderstand attribute can be used to indicate whether a header entry is mandatory or optional for the recipient to process.

If you add mustUnderstand="1" to a child element of the Header element it indicates that the receiver processing the Header must recognize the element. If the receiver does not recognize the element it will fail when processing the Header.

### Syntax

soap:mustUnderstand="0|1"

### Example

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Header>  
  <m:Trans xmlns:m="http://www.w3schools.com/transaction/"  
  soap:mustUnderstand="1">234  
  </m:Trans>  
</soap:Header>  
...  
...  
</soap:Envelope>

## The actor Attribute

A SOAP message may travel from a sender to a receiver by passing different endpoints along the message path. However, not all parts of a SOAP message may be intended for the ultimate endpoint, instead, it may be intended for one or more of the endpoints on the message path.

The SOAP actor attribute is used to address the Header element to a specific endpoint.

### Syntax

soap:actor="*URI*"

### Example

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Header>  
  <m:Trans xmlns:m="http://www.w3schools.com/transaction/"  
  soap:actor="http://www.w3schools.com/appml/">234  
  </m:Trans>  
</soap:Header>  
...  
...  
</soap:Envelope>

## The encodingStyle Attribute

The encodingStyle attribute is used to define the data types used in the document. This attribute may appear on any SOAP element, and it will apply to that element's contents and all child elements.

A SOAP message has no default encoding.

### Syntax

soap:encodingStyle="*URI*"

## The SOAP Body Element

The required SOAP Body element contains the actual SOAP message intended for the ultimate endpoint of the message.

Immediate child elements of the SOAP Body element may be namespace-qualified.

### Example

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Body>  
  <m:GetPrice xmlns:m="http://www.w3schools.com/prices">  
    <m:Item>Apples</m:Item>  
  </m:GetPrice>  
</soap:Body>  
  
</soap:Envelope>

The example above requests the price of apples. Note that the m:GetPrice and the Item elements above are application-specific elements. They are not a part of the SOAP namespace.

A SOAP response could look something like this:

<?xml version="1.0"?>  
  
<soap:Envelope  
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"  
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">  
  
<soap:Body>  
  <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">  
    <m:Price>1.90</m:Price>  
  </m:GetPriceResponse>  
</soap:Body>  
  
</soap:Envelope>

WSDL stands for Web Services Description Language. It is the standard format for describing a web service. WSDL was developed jointly by Microsoft and IBM.

## Features of WSDL

* WSDL is an XML-based protocol for information exchange in decentralized and distributed environments.
* WSDL definitions describe how to access a web service and what operations it will perform.
* WSDL is a language for describing how to interface with XML-based services.
* WSDL is an integral part of Universal Description, Discovery, and Integration (UDDI), an XML-based worldwide business registry.
* WSDL is the language that UDDI uses.
* WSDL is pronounced as 'wiz-dull' and spelled out as 'W-S-D-L'.

## WSDL Usage

WSDL is often used in combination with SOAP and XML Schema to provide web services over the Internet. A client program connecting to a web service can read the WSDL to determine what functions are available on the server. Any special datatypes used are embedded in the WSDL file in the form of XML Schema. The client can then use SOAP to actually call one of the functions listed in the WSDL.

The three major elements of WSDL that can be defined separately are:

* Types
* Operations
* Binding

A WSDL document has various elements, but they are contained within these three main elements, which can be developed as separate documents and then they can be combined or reused to form complete WSDL files.

WSDL Elements

A WSDL document contains the following elements:

* **Definition** : It is the root element of all WSDL documents. It defines the name of the web service, declares multiple namespaces used throughout the remainder of the document, and contains all the service elements described here.
* **Data types** : The data types to be used in the messages are in the form of XML schemas.
* **Message** : It is an abstract definition of the data, in the form of a message presented either as an entire document or as arguments to be mapped to a method invocation.
* **Operation** : It is the abstract definition of the operation for a message, such as naming a method, message queue, or business process, that will accept and process the message.
* **Port type** : It is an abstract set of operations mapped to one or more end-points, defining the collection of operations for a binding; the collection of operations, as it is abstract, can be mapped to multiple transports through various bindings.
* **Binding** : It is the concrete protocol and data formats for the operations and messages defined for a particular port type.
* **Port** : It is a combination of a binding and a network address, providing the target address of the service communication.
* **Service** : It is a collection of related end-points encompassing the service definitions in the file; the services map the binding to the port and include any extensibility definitions.

In addition to these major elements, the WSDL specification also defines the following utility elements:

* **Documentation:** This element is used to provide human-readable documentation and can be included inside any other WSDL element.
* **Import** : This element is used to import other WSDL documents or XML Schemas.

**NOTE:** WSDL parts are usually generated automatically using web services-aware tools.

The WSDL Document Structure

The main structure of a WSDL document looks like this:

<definitions>

<types>

definition of types........

</types>

<message>

definition of a message....

</message>

<portType>

<operation>

definition of a operation.......

</operation>

</portType>

<binding>

definition of a binding....

</binding>

<service>

definition of a service....

</service>

</definitions>

A WSDL document can also contain other elements, like extension elements and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

UDDI is an XML-based standard for describing, publishing, and finding web services.

* UDDI stands for **Universal Description, Discovery, and Integration.**
* UDDI is a specification for a distributed registry of web services.
* UDDI is a platform-independent, open framework.
* UDDI can communicate via SOAP, CORBA, Java RMI Protocol.
* UDDI uses Web Service Definition Language(WSDL) to describe interfaces to web services.
* UDDI is seen with SOAP and WSDL as one of the three foundation standards of web services.
* UDDI is an open industry initiative, enabling businesses to discover each other and define how they interact over the Internet.

UDDI has two sections:

* A registry of all web service's metadata, including a pointer to the WSDL description of a service.
* A set of WSDL port type definitions for manipulating and searching that registry.