**Soap web service** – Simple Object Access Protocol

is a messaging [protocol](https://en.wikipedia.org/wiki/Protocol_(computing)) specification for exchanging structured information in the implementation of [web services](https://en.wikipedia.org/wiki/Web_service) in [computer networks](https://en.wikipedia.org/wiki/Computer_network). Its purpose is to induce [extensibility](https://en.wikipedia.org/wiki/Extensibility), [neutrality](https://en.wikipedia.org/wiki/Neutrality_(philosophy)) and independence. It uses [XML Information Set](https://en.wikipedia.org/wiki/XML_Information_Set) for its [message format](https://en.wikipedia.org/wiki/Message_format), and relies on [application layer](https://en.wikipedia.org/wiki/Application_layer) protocols, most often [Hypertext Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP) or [Simple Mail Transfer Protocol](https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) (SMTP), for message negotiation and transmission.

SOAP allows clients to invoke web services and receive responses independent of language and platforms.

* an envelope, which defines the message structure[[1]](https://en.wikipedia.org/wiki/SOAP#cite_note-1) and how to process it
* a set of encoding rules for expressing instances of application-defined datatypes
* a convention for representing procedure calls and responses

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

**ELEMENTS**

**Envelope**(required) −This is the root element and encapsulates the complete information.

**Header**(Optional) − Holds some attributes of the message that are helpful in processing the message

**Body**(required) − It holds the actual XML data that we are intersted in sendin accross the network.

**Fault**(optional) − provieds error information that may occurs during the processing of the message.

<https://www.w3schools.com/xml/xml_soap.asp>

**WSDL**

* WSDL stands for Web Services Description Language
* WSDL is used to describe web services
* WSDL is written in XML

|  |  |
| --- | --- |
| **Element** | **Description** |
| <types> | Defines the (XML Schema) data types used by the web service |
| <message> | Defines the data elements for each operation |
| <portType> | Describes the operations that can be performed and the messages involved. |
| <binding> | Defines the protocol and data format for each port type |

## WSDL Binding to SOAP

The**binding** element has two attributes - name and type.

The name attribute (you can use any name you want) defines the name of the binding, and the type attribute points to the port for the binding, in this case the "glossaryTerms" port.

The **soap:binding** element has two attributes - style and transport.

The style attribute can be "rpc" or "document". In this case we use document. The transport attribute defines the SOAP protocol to use. In this case we use HTTP.

The**operation** element defines each operation that the portType exposes.

For each operation the corresponding SOAP action has to be defined. You must also specify how the input and output are encoded. In this case we use "literal".

<https://www.w3schools.com/xml/xml_wsdl.asp>

XML

XML stands for eXtensible Markup Language.

XML was designed to store and transport data.

XML was designed to be both human- and machine-readable.

But still, the XML does not DO anything. XML is just information wrapped in tags.

XML and HTML were designed with different goals:

* XML was designed to carry data - with focus on what data is
* HTML was designed to display data - with focus on how data looks
* XML tags are not predefined like HTML tags are

## XML Does Not Use Predefined Tags

The XML language has no predefined tags.

The tags in the example above (like <to> and <from>) are not defined in any XML standard. These tags are "invented" by the author of the XML document.

HTML works with predefined tags like <p>, <h1>, <table>, etc.

XML does not carry any information about how to be displayed.

In many HTML applications, XML is used to store or transport data, while HTML is used to format and display the same data.

When displaying data in HTML, you should not have to edit the HTML file when the data changes.

With XML, the data can be stored in separate XML files.

With a few lines of JavaScript code, you can read an XML file and update the data content of any HTML page.

<https://www.w3schools.com/xml/xml_usedfor.asp>

**XSD**

The XML Schema language is also referred to as XML Schema Definition (XSD)

## XML Schemas Support Data Types

## XML Schemas use XML Syntax

## Ukratko, XML, samo što postoje tipovi podataka i onda definiramo u toj shemi što je koji tip podatka.. To se u zaglavlju određuje, ajmo reći da je XSD XML-u isto što je CSS HTML-u

## https://www.w3schools.com/xml/schema\_schema.asp

Rest web service - REpresentational State Transfer

REST is web standards based architecture and uses HTTP Protocol. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods

HTTP methods:

* **GET** − Provides a read only access to a resource.
* **POST** − Used to create a new resource.
* **DELETE** − Used to remove a resource.
* **PUT** − Used to update a existing resource or create a new resource

REST ide sa JSON-om iako može i sa XML-om a SOAP samo XML.

**https://en.wikipedia.org/wiki/Representational\_state\_transfer**

JSON

JSON: **J**ava**S**cript **O**bject **N**otation.

JSON is a syntax for storing and exchanging data.

JSON is text, written with JavaScript object notation.

JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server.

JavaScript has a built in function to convert a string, written in JSON format, into native JavaScript objects:

JSON.parse()

In **JSON**, *values* must be one of the following data types:

* a string
* a number
* an object (JSON object)
* an array
* a boolean
* null

## JSON Uses JavaScript Syntax

Both JSON and XML can be used to receive data from a web server.

JSON is Like XML Because

* Both JSON and XML are "self describing" (human readable)
* Both JSON and XML are hierarchical (values within values)
* Both JSON and XML can be parsed and used by lots of programming languages
* Both JSON and XML can be fetched with an XMLHttpRequest

JSON is Unlike XML Because

* JSON doesn't use end tag
* JSON is shorter
* JSON is quicker to read and write
* JSON can use arrays

The biggest difference is:

 XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

Why JSON is Better Than XML

XML is much more difficult to parse than JSON.  
JSON is parsed into a ready-to-use JavaScript object.

For AJAX applications, JSON is faster and easier than XML:

Using XML

* Fetch an XML document
* Use the XML DOM to loop through the document
* Extract values and store in variables

Using JSON

* Fetch a JSON string
* JSON.Parse the JSON string

https://www.w3schools.com/js/js\_json\_intro.asp