

# XML Soap

[< Previous](#)[Next >](#)

- 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/>

[Dark mode](#)

# 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 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.

```
<?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
```

Dark mode

```
...  
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="https://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

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="https://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="https://www.w3schools.com/transaction/"
      soap:actor="https://www.w3schools.com/code/">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.

# 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="https://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="https://www.w3schools.com/prices">
      <m:Price>1.90</m:Price>
    </m:GetPriceResponse>
  </soap:Body>

</soap:Envelope>
```

## The SOAP Fault Element

The SOAP Fault element holds errors and status information for a SOAP message.

If a Fault element is present, it must appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.

The SOAP Fault element has the following sub elements:

Sub Element	Description
<faultcode>	A code for identifying the fault
<faultstring>	A human readable explanation of the fault
<faultactor>	Information about who caused the fault to happen
<detail>	Holds application specific error information related to the Body element

## SOAP Fault Codes

The faultcode values defined below must be used in the faultcode element when describing faults:

Error	Description
VersionMismatch	Found an invalid namespace for the SOAP Envelope element
MustUnderstand	An immediate child element of the Header element, with the mustUnderstand attribute set to "1", was not understood
Client	The message was incorrectly formed or contained incorrect information
Server	There was a problem with the server so the message could not proceed

## The HTTP Protocol

HTTP communicates over TCP/IP. An HTTP client connects to an HTTP server using TCP. After establishing a connection, the client can send an HTTP request message to the server:

```
POST /item HTTP/1.1
Host: 189.123.255.239
```



The server then processes the request and sends an HTTP response back to the client. The response contains a status code that indicates the status of the request:

```
200 OK
Content-Type: text/plain
Content-Length: 200
```

In the example above, the server returned a status code of 200. This is the standard success code for HTTP.

If the server could not decode the request, it could have returned something like this:

```
400 Bad Request
Content-Length: 0
```

## SOAP Binding

The SOAP specification defines the structure of the SOAP messages, not how they are exchanged. This gap is filled by what is called "SOAP Bindings". SOAP bindings are mechanisms which allow SOAP messages to be effectively exchanged using a transport protocol.

Most SOAP implementations provide bindings for common transport protocols, such as HTTP or SMTP.

HTTP is synchronous and widely used. A SOAP HTTP request specifies at least two HTTP headers: Content-Type and Content-Length.

SMTP is asynchronous and is used in last resort or particular cases.

Java implementations of SOAP usually provide a specific binding for the JMS (Java Messaging System) protocol.

## Content-Type

The Content-Type header for a SOAP request and response defines the MIME type for the message and the character encoding (optional) used for the XML body of the re

## Syntax

```
Content-Type: MIMEType; charset=character-encoding
```

## Example

```
POST /item HTTP/1.1  
Content-Type: application/soap+xml; charset=utf-8
```

# Content-Length

The Content-Length header for a SOAP request and response specifies the number of bytes in the body of the request or response.

## Syntax

```
Content-Length: bytes
```

## Example

```
POST /item HTTP/1.1  
Content-Type: application/soap+xml; charset=utf-8  
Content-Length: 250
```

# A SOAP Example

In the example below, a GetStockPrice request is sent to a server. The request has a StockName parameter, and a Price parameter that will be returned in the response. The namespace for the function is defined in "http://www.example.org/stock".

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?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 xmlns:m="http://www.example.org/stock">
    <m:GetStockPrice>
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>

</soap:Envelope>
```

## The SOAP response:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>

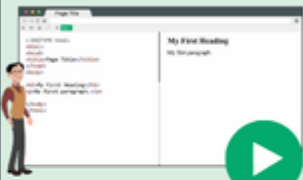
<soap:Envelope
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">

  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPriceResponse>
      <m:Price>34.5</m:Price>
    </m:GetStockPriceResponse>
  </soap:Body>

</soap:Envelope>
```


[HTML](#)[CSS](#)[JAVASCRIPT](#)

## Visual Learner?



Be taught by an instructor through our video tutorial!

[WATCH NOW](#)



COLOR PICKER



Get certified  
by completing  
a XML  
course today!



[Get started](#)

Dark mode

### Learning Journey

- ✓ No More Ads
- ✓ Hosting Benefits
- ✓ More Spaces
- ✓ HTML Video

UPGRADE



Report Error

Spaces

Upgrade

Newsletter

Get Certified

### Top Tutorials

HTML Tutorial  
CSS Tutorial  
JavaScript Tutorial  
How To Tutorial  
SQL Tutorial  
Python Tutorial  
W3.CSS Tutorial  
Bootstrap Tutorial  
PHP Tutorial  
Java Tutorial  
C++ Tutorial  
jQuery Tutorial

### Top References

HTML Reference  
CSS Reference  
JavaScript Reference  
SQL Reference  
Python Reference  
W3.CSS Reference

Dark mode

#### HTML COURS

[Java Reference](#)  
[Angular Reference](#)  
[jQuery Reference](#)

### Top Examples

[HTML Examples](#)  
[CSS Examples](#)  
[JavaScript Examples](#)  
[How To Examples](#)  
[SQL Examples](#)  
[Python Examples](#)  
[W3.CSS Examples](#)  
[Bootstrap Examples](#)  
[PHP Examples](#)  
[Java Examples](#)  
[XML Examples](#)  
[jQuery Examples](#)

### Get Certified

[HTML Certificate](#)  
[CSS Certificate](#)  
[JavaScript Certificate](#)  
[Front End Certificate](#)  
[SQL Certificate](#)  
[Python Certificate](#)  
[PHP Certificate](#)  
[jQuery Certificate](#)  
[Java Certificate](#)  
[C++ Certificate](#)  
[C# Certificate](#)  
[XML Certificate](#)

---

[FORUM](#) | [ABOUT](#)

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our [terms of use](#), [cookie and privacy policy](#).

Copyright 1999-2022 by Refsnes Data. All Rights Reserved.  
W3Schools is Powered by W3.CSS.

