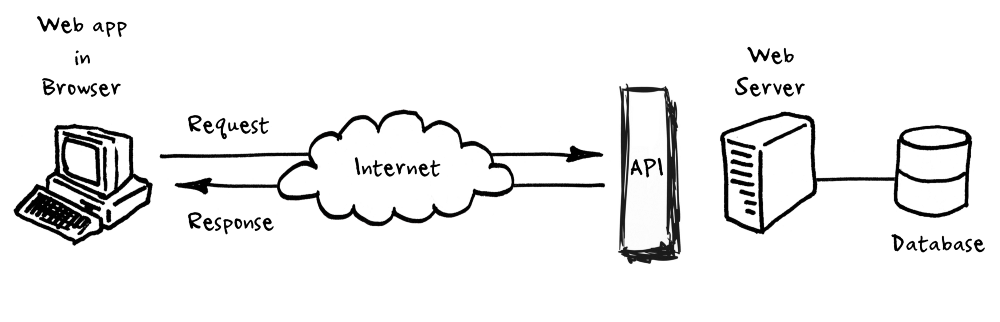
**API (Application Programming Interface)**

API is used to allow the application to communicate with one another with the set of rules. When people speak of “an API”, they sometimes generalize and actually mean “a publicly available web-based API that returns data, likely in JSON or XML”. The API is not the database or even the server, it is the code that governs the *access point(s)* for the server.



They allow us to go get data from outside sources.

1. We can send an API a request detailing the information we want.
2. APIs allow our sites to alter data on other applications, too. For instance, you’ve probably seen “Share on Facebook” or “Share on Twitter” buttons on miscellaneous websites. When/if you click one of these buttons, the site you’re visiting can communicate with your Facebook or Twitter account, and alter its data by adding new status or tweet.

Main types of Web APIs

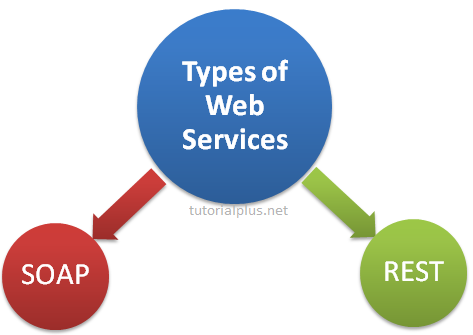
There are four main types of APIs:

* Open APIs: Also known as Public API, there are no restrictions to access these types of APIs because they are publicly available.
* Partner APIs: One needs specific rights or licenses in order to access this type of API because they are not available to the public.
* Internal APIs: Also known as Private APIs, only internal systems expose this type of API, which is, therefore, less known and often meant for use inside the company.  The company uses this type of API among the different internal teams to be able to improve its products and services.
* Composite APIs: This type of API combines different data and service APIs. It is a sequence of tasks that run synchronously as a result of the execution and not at the request of a task. Its main uses are to speed up the process of execution and improve the performance of the listeners in the web interfaces.

## Web service APIs

Apart from the main web APIs, there are also web service APIs:

* [SOAP](https://blog.rapidapi.com/soap-vs-rest-api/)
* XML-RPC
* [JSON](https://blog.rapidapi.com/api-glossary/json/)-RPC
* REST



A web service is a system or software that uses an address, i.e., URL on the World Wide Web, to provide access to its services.

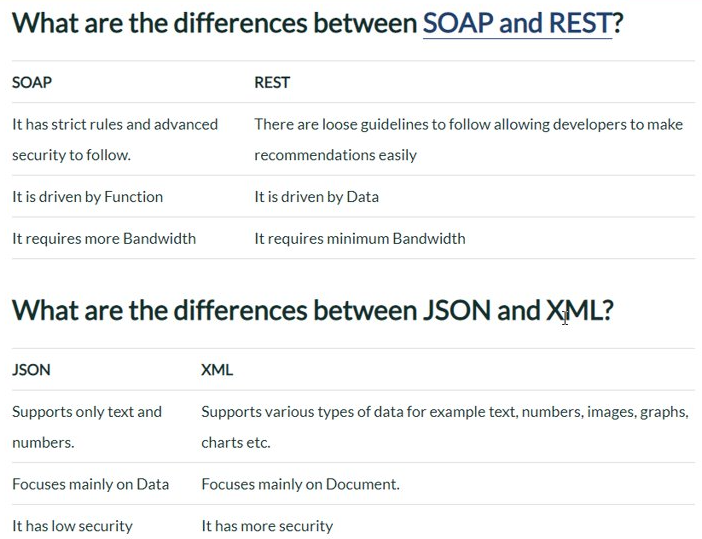
The following are the most common types of web service APIs:

* SOAP (Simple Object Access Protocol): This is a protocol that uses XML as a format to transfer data. Its main function is to define the structure of the messages and method of communication. It also uses WSDL, or Web Services Definition Language, in a machine-readable document to publish a definition of its interface.
* XML-RPC: This is a protocol that uses a specific XML format to transfer data compared to SOAP that uses a proprietary XML format. It is also older than SOAP. XML-RPC uses minimum bandwidth and is much simpler than SOAP. Example



* REST (Representational State Transfer): REST is not a protocol like the other web services, instead, it is a set of architectural principles. The REST service needs to have certain characteristics, including simple interfaces, which are resources identified easily within the request and manipulation of resources using the interface.
* JSON-RPC: This protocol is similar to XML-RPC but instead of using XML format to transfer data it uses JSON. Example



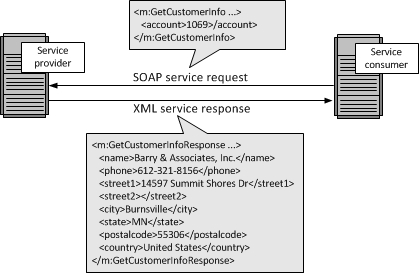


## SOAP

Stands for "Simple Object Access Protocol," and can do more than just get your hands clean. SOAP is a method of transferring messages, or small amounts of information, over the Internet. SOAP is a W3C recommendation for communication between two applications.

SOAP is XML based protocol. It is platform independent and language independent. By using SOAP, you will be able to interact with other programming language applications. SOAP messages are formatted in [XML](https://techterms.com/definition/xml) and are typically sent using [HTTP](https://techterms.com/definition/http) (hypertext transfer protocol). Both are widely supported data transmission standards. HTTP, which is the protocol that Web pages are sent over, has the additional advantage of avoiding most network [firewalls](https://techterms.com/definition/firewall). Since firewalls usually do not block port 80 (HTTP) traffic, most SOAP messages can pass through without any problems.

Each SOAP message is contained in an "envelope" that includes a header and a body. The header may include the message ID and date the message was sent, while the body contains the actual message. Because SOAP messages all use the same format, they are compatible with many different operating systems and protocols. For example, a user can send a SOAP message from a Windows XP machine to a Unix-based Web server without worrying about the message being altered. The Unix machine can then redirect the message to the appropriate location or open the file using a program on the system. While most SOAP messages are sent over the Web via HTTP, they can also be sent via e-mail, using [SMTP](https://techterms.com/definition/smtp).



# **WSDL**

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.

## History of WSDL

WSDL 1.1 was submitted as a W3C Note by Ariba, IBM, and Microsoft for describing services for the W3C XML Activity on XML Protocols in March 2001.

WSDL 1.1 has not been endorsed by the World Wide Web Consortium (W3C), however it has just released a draft for version 2.0 that will be a recommendation (an official standard), and thus endorsed by the W3C.

WSDL breaks down web services into three specific, identifiable elements that can be combined or reused once defined.

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 −



## What is WSDL?

* **WSDL stands for Web Services Description Language**
* **WSDL is written in XML**
* **WSDL is an XML document**
* **WSDL is used to describe Web services**
* **WSDL is also used to locate Web services**
* **WSDL is a W3C recommendation**

## WSDL Describes Web Services

**WSDL stands for Web Services Description Language.**

**WSDL is a document written in XML. The document describes a Web service. It specifies the location of the service and the operations (or methods) the service exposes.**

## WSDL is a W3C Recommendation

**WSDL became a W3C Recommendation 26. June 2007.**

**The WSDL Document Structure**

**A WSDL document describes a web service using these major elements:**

|  |  |
| --- | --- |
| **Element** | **Defines** |
| **<portType>** | **The operations performed by the web service** |
| **<message>** | **The messages used by the web service** |
| **<types>** | **The data types used by the web service** |
| **<binding>** | **The communication protocols used by the web service** |

**The main structure of a WSDL document looks like this:**

****

**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.**

**For a complete syntax overview go to the chapter**[**WSDL Syntax**](http://w3schools.sinsixx.com/wsdl/wsdl_syntax.asp.htm)**.**

**WSDL Ports**

**The <portType> element is the most important WSDL element.**

**It describes a web service, the operations that can be performed, and the messages that are involved.**

**The <portType> element can be compared to a function library (or a module, or a class) in a traditional programming language.**

**WSDL Messages**

**The <message> element defines the data elements of an operation.**

**Each message can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language.**

**WSDL Types**

**The <types> element defines the data type that are used by the web service.**

**For maximum platform neutrality, WSDL uses XML Schema syntax to define data types.**

**WSDL Bindings**

**The <binding> element defines the message format and protocol details for each port.**

## Advantages of Soap Web Services

* WS Security: SOAP defines its own security known as WS Security.
* Language and Platform independent: SOAP web services can be written in any programming language and executed in any platform.

## Disadvantages of Soap Web Services

* Slow: SOAP uses XML format that must be parsed to be read. It defines many standards that must be followed while developing the SOAP applications. So it is slow and consumes more bandwidth and resource.
* WSDL dependent: SOAP uses WSDL and doesn't have any other mechanism to discover the service.

## SOAP Advantages and Disadvantages

**SOAP is an integral part of the service-oriented architecture (**[**SOA**](https://searchmicroservices.techtarget.com/definition/service-oriented-architecture-SOA)**) and the**[**Web services**](https://searchmicroservices.techtarget.com/definition/Web-services-application-services)**specifications associated with SOA. Because it allows the sender to create a message route based on the logical services that have to be applied to the message on the way to its destination, it lends itself to providing secure and compliant connections, controlling access, offering reliable delivery and failure recovery, and supporting dynamic service discovery. SOA without SOAP is difficult to imagine.**

**SOAP’s messages are defined at a high level in**[**XML**](https://searchmicroservices.techtarget.com/definition/XML-Extensible-Markup-Language)**, but most SOAP applications use Web Services Definition Language (**[**WSDL**](https://searchmicroservices.techtarget.com/definition/Web-Services-Description-Language-WSDL)**), which is authored in XML.  The XML structure of SOAP makes it handy for applications that expect their information to be provided in XML form, and the fact that SOAP can ride on a variety of network protocols, including HTTP, means it’s easily passed through firewalls, where other protocols might require special accommodation.**

**The data structure of SOAP is based on XML, which is similar in many ways to the HTML used to define web pages. Like HTML, XLM is largely human-readable, which makes it fairly easy to understand a SOAP message, but also makes the messages relatively large in comparison to the Common Object Request Broker Architecture (**[**CORBA**](https://searchsqlserver.techtarget.com/definition/CORBA)**) and its Remote Procedure Call (**[**RPC**](https://searchmicroservices.techtarget.com/definition/Remote-Procedure-Call-RPC)**) protocol that will accommodate binary data.**

**The biggest disadvantage of SOAP (and SOA overall) is that it’s a heavyweight protocol for a heavyweight architecture.  The notion of a message passing through a string of nodes to be processed by each seems to mix protocols and service bus architectural models for software, and neither of those two are considered optimal for microservice-based development as popularly used today.**

## UDDI

UDDI uses World Wide Web Consortium (W3C) and Internet Engineering Task Force (IETF) Internet standards such as XML, HTTP, and DNS protocols.

UDDI uses WSDL to describe interfaces to web services

Additionally, cross platform programming features are addressed by adopting SOAP, known as XML Protocol messaging specifications found at the W3C Web site.

## UDDI Benefits

Any industry or businesses of all sizes can benefit from UDDI

Before UDDI, there was no Internet standard for businesses to reach their customers and partners with information about their products and services. Nor was there a method of how to integrate into each other's systems and processes.   
  
Problems the UDDI specification can help to solve:

* Making it possible to discover the right business from the millions currently online
* Defining how to enable commerce once the preferred business is discovered
* Reaching new customers and increasing access to current customers
* Expanding offerings and extending market reach
* Solving customer-driven need to remove barriers to allow for rapid participation in the global Internet economy
* Describing services and business processes programmatically in a single, open, and secure environment

## How can UDDI be Used

If the industry published an UDDI standard for flight rate checking and reservation, airlines could register their services into an UDDI directory. Travel agencies could then search the UDDI directory to find the airline's reservation interface. When the interface is found, the travel agency can communicate with the service immediately because it uses a well-defined reservation interface.

## Who is Supporting UDDI?

UDDI is a cross-industry effort driven by all major platform and software providers like Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP, and Sun, as well as a large community of marketplace operators, and e-business leaders.

Over 220 companies are members of the UDDI community.

# Introduction to XML

[home](http://w3schools.sinsixx.com/xml/default.asp.htm) [next](http://w3schools.sinsixx.com/xml/xml_usedfor.asp.htm)

**XML was designed to transport and store data.**

**HTML was designed to display data.**

## What You Should Already Know

Before you continue you should have a basic understanding of the following:

* HTML
* JavaScript

If you want to study these subjects first, find the tutorials on our [Home page](http://w3schools.sinsixx.com/default.asp.htm).

## What is XML?

* XML stands for EXtensible Markup Language
* XML is a markup language much like HTML
* XML was designed to carry data, not to display data
* XML tags are not predefined. You must define your own tags
* XML is designed to be self-descriptive
* XML is a W3C Recommendation

## The Difference Between XML and HTML

XML is not a replacement for HTML.

XML and HTML were designed with different goals:

* XML was designed to transport and store data, with focus on what data is.
* HTML was designed to display data, with focus on how data looks.

HTML is about displaying information, while XML is about carrying information.

## XML Does not DO Anything

Maybe it is a little hard to understand, but XML does not DO anything. XML was created to structure, store, and transport information.

The following example is a note to Tove from Jani, stored as XML:

|  |
| --- |
| <note>  <to>Tove</to>  <from>Jani</from>  <heading>Reminder</heading>  <body>Don't forget me this weekend!</body>  </note> |

The note above is quite self descriptive. It has sender and receiver information, it also has a heading and a message body.

But still, this XML document does not DO anything. It is just pure information wrapped in tags. Someone must write a piece of software to send, receive or display it.

## XML is Just Plain Text

XML is nothing special. It is just plain text. Software that can handle plain text can also handle XML.

However, XML-aware applications can handle the XML tags specially. The functional meaning of the tags depends on the nature of the application.

## With XML You Invent Your Own 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.

That is because the XML language has no predefined tags.

The tags used in HTML (and the structure of HTML) are predefined. HTML documents can only use tags defined in the HTML standard (like <p>, <h1>, etc.).

XML allows the author to define his own tags and his own document structure.

## XML is Not a Replacement for HTML

XML is a complement to HTML.

It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data.

My best description of XML is this:

**XML is a software and hardware independent tool for carrying information.**

## XML is a W3C Recommendation

XML became a W3C Recommendation 10. February 1998.

To read more about the XML activities at W3C, please read our [W3C Tutorial](http://w3schools.sinsixx.com/w3c/w3c_xml.asp.htm).

## XML is Everywhere

We have been participating in XML development since its creation. It has been amazing to see how quickly the XML standard has developed, and how quickly a large number of software vendors have adopted the standard.

XML is now as important for the Web as HTML was to the foundation of the Web.

XML is everywhere. It is the most common tool for data transmissions between all sorts of applications, and is becoming more and more popular in the area of storing and describing information.

# How Can XML be Used?

[previous](http://w3schools.sinsixx.com/xml/xml_whatis.asp.htm) [next](http://w3schools.sinsixx.com/xml/xml_tree.asp.htm)

**XML is used in many aspects of web development, often to simplify data storage and sharing.**

## XML Separates Data from HTML

If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes.

With XML, data can be stored in separate XML files. This way you can concentrate on using HTML for layout and display, and be sure that changes in the underlying data will not require any changes to the HTML.

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

**You will learn more about this in a later chapter of this tutorial.**

## XML Simplifies Data Sharing

In the real world, computer systems and databases contain data in incompatible formats.

XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data.

This makes it much easier to create data that different applications can share.

## XML Simplifies Data Transport

With XML, data can easily be exchanged between incompatible systems.

One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet.

Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications.

## XML Simplifies Platform Changes

Upgrading to new systems (hardware or software platforms), is always very time consuming. Large amounts of data must be converted and incompatible data is often lost.

XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

## XML Makes Your Data More Available

Since XML is independent of hardware, software and application, XML can make your data more available and useful.

Different applications can access your data, not only in HTML pages, but also from XML data sources.

With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc), and make it more available for blind people, or people with other disabilities.

## XML is Used to Create New Internet Languages

A lot of new Internet languages are created with XML.

Here are some examples:

* XHTML the latest version of HTML
* WSDL for describing available web services
* WAP and WML as markup languages for handheld devices
* RSS languages for news feeds
* RDF and OWL for describing resources and ontology
* SMIL for describing multimedia for the web