API

The term web API generally refers to both sides of computer systems communicating over a network: the API services offered by a server, as well as the API offered by the client such as a web browser.

The server-side portion of the web API is a programmatic interface to a defined request-response message system and is typically referred to as the Web Service. There are several design models for web services, but the two most dominant are SOAP and REST.

What Is a SOAP API?

SOAP is a standard communication protocol system that permits processes using different operating systems like Linux and Windows to communicate via HTTP and its XML. SOAP based APIs are designed to create, recover, update and delete records like accounts, passwords, leads, and custom objects.

These offers over twenty different kinds of calls that make it easy for the API developers to maintain their accounts, perform accurate searches and much more. These can then be used with all those languages that support web services.

SOAP APIs take the advantages of making web-based protocols such as HTTP and its XML that are already operating the all operating systems that are why its developers can easily manipulate web services and get responses without caring about language and platforms at all.

Example:

A sample message exchange looks like the following.

```
POST http://www.stgregorioschurchdc.org/cgi/websvccal.cgi HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: "http://www.stgregorioschurchdc.org/Calendar#easter_date"
Content-Length: 479
Host: www.stgregorioschurchdc.org
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
<?xml version="1.0"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cal="http://www.stgregorioschurchdc.org/Calendar">
<soapenv:Header/>
<soapenv:Body>
   <cal:easter_date soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <year xsi:type="xsd:short">2014</year>
</cal:easter_date>
</soapenv:Body>
</soapenv:Envelope>
```

The response from the service:

```
HTTP/1.1 200 OK
Date: Fri, 22 Nov 2013 21:09:44 GMT
Server: Apache/2.0.52 (Red Hat)
SOAPServer: SOAP::Lite/Perl/0.52
Content-Length: 566
Connection: close
Content-Type: text/xml; charset=utf-8
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
   <namesp1:easter_dateResponse
xmlns:namesp1="http://www.stgregorioschurchdc.org/Calendar">
<s-gensym3 xsi:type="xsd:string">2014/04/20</s-gensym3>
</namesp1:easter_dateResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

From this example we can see the message was sent over HTTP. SOAP is agnostic of the underlying transport protocol and can be sent over almost any protocol such as HTTP, SMTP, TCP, or JMS. As was already mentioned, the SOAP message itself must be XML-formatted. As is normal for any XML document, there must be one root element: The Envelope in this case.

This contains two required elements: The Header and the Body. The rest of the elements in this message are described by the WSDL. The accompanying WSDL that defines the above service looks like this (the details are not important, but the entire document is shown here for completeness):

WSDL:

WSDL is Web Service Description Language, is an XML based definition language. It's used for describing the functionality of a **SOAP** based web service. **WSDL** files are central to testing **SOAP**-based services. SoapUI uses **WSDL** files to generate test requests, assertions and mock services.

What WSDL file contains?

A WSDL document has a definitions element that contains the other five elements, types, message, portType, binding and service. The following sections describe the features of the generated client code.

What Is a REST API?

REST is basically an architectural style of the web services that work as a channel of communication between different computers or systems on the internet. The term REST API is something else.

Those application programming interfaces that are backed by the architectural style of REST architectural system are called REST APIs. REST API compliant web services, database systems, and computer systems permit requesting systems to get robust

access and redefine representations of web-based resources by deploying a predefined set of stateless protocols and standard operations.

By these protocols and operations and redeploying the manageable and updatable components without causing the effect on the system, REST API systems deliver fast performance, reliability, and more progression.

GET	Read or retrieve data
POST	Add new data
PUT	Update data that already exists
DELETE	Remove data

Example:

A sample message exchange could contain as little as this -

Request:

```
GET http://www.catechizeme.com/catechisms/catechism_for_young_children/daily_question.js HTTP/1.1
Accept-Encoding: gzip,deflate
Host: www.catechizeme.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 22 Nov 2013 22:32:22 GMT
Server: Apache
X-Powered-By: Phusion Passenger (mod_rails/mod_rack) 3.0.17
ETag: "b8a7ef8b4b282a70d1b64ea5e79072df"
X-Runtime: 13
Cache-Control: private, max-age=0, must-revalidate
Content-Length: 209
Status: 200
Keep-Alive: timeout=2, max=100
Connection: Keep-Alive
Content-Type: js; charset=utf-8
{
   "link": "catechisms\/catechism_for_young_children\/questions\/36",
   "catechism": "Catechism for Young Children",
   "a": "Original sin.",
   "position": 36,
   "q": " What is that sinful nature which we inherit from Adam called?"
}
```

As is already expected this message was sent over HTTP and used the GET verb.

Further note that the URI, which also had to be included in the SOAP request, but there it had no meaning, here actually takes on a meaning. The body of the message is significantly smaller, in this example there actually isn't one.

Differences:

- REST API has no official standard at all because it is an architectural style. SOAP API, has an official standard because it is a protocol.
- REST APIs uses multiple standards like HTTP, JSON, URL, and XML while SOAP APIs is largely based on HTTP and XML.
- As REST API deploys multiple standards, so it takes fewer resources and bandwidth as compared to SOAP that uses XML for the creation of Payload and results in the large sized file.
- The ways both APIs exposes the business logics are also different. REST API takes advantage of URL exposure like @path("/WeatherService") while SOAP API use of services interfaces like @WebService.
- SOAP API defines too many standards, and its implementer implements the things in a standard way only. In the case of miscommunication from service, the result will be the error. REST API, on the other hand, don't make emphasis on too many standards and results in corrupt API in the end.
- REST API uses Web Application Description Language, and SOAP API used Web Services Description language for describing the functionalities being offered by web services.
- REST APIs are more convenient with JavaScript and can be implemented easily as well. SOAP APIs are also convenient with JavaScript but don't support for greater implementation.

DNS: The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

HTTP: hypertext transfer protocol, hypertext transfer protocol: the standard protocol for transferring hypertext documents on the World Wide Web.

HTTPS: HTTPS (Hypertext Transfer Protocol Secure) is an internet communication protocol that protects the integrity and confidentiality of data between the user's computer and the site. Users expect a secure and private online experience when using a website.

SSL: secure sockets layer, SSL Stands for secure sockets layer. Protocol for web browsers and servers that allows for the authentication, encryption and decryption of data sent over the Internet. ... Wildcard SSL certificates Type of certificate used to secure multiple subdomains.

TLS: Transport Layer Security,Transport Layer Security (TLS) is the successor protocol to SSL. TLS is an improved version of SSL. It works in much the same way as the SSL, using encryption to protect the transfer of data and information. The two terms are often used interchangeably in the industry although SSL is still widely used.

FTP: File transfer protocol, File transfer protocol (FTP) is a set of rules that computers follow for the transferring of files from one system to another over the internet. It may be used by a business to transfer files from one computer system to another, or websites may use FTP to upload or download files from a website's server.

SFTP: SFTP (SSH File Transfer Protocol, also known as Secure FTP) is a popular method for securely transferring files over remote systems. SFTP was designed as an extension of the Secure Shell protocol (SSH) version 2.0 to enhance secure file transfer capabilities.

SSH: Secure Shell (**SSH**) is a cryptographic network protocol for operating network services securely over an unsecured network ,SSH is typically used to log into a remote machine and execute commands, but it also supports tunneling, forwarding TCP ports and X11 connections; it can transfer files using the associated SSH file transfer (SFTP) or secure copy (SCP) protocols. SSH uses the client-server model.