We have used Web Services, ASMX and SVC, the difference is:

* ASMX is simpler, but and old technology.
* SVC is more complex to develop, but this is the newer technology that is going to be supported.

And what about **Web Services and Web API 2.0**  which are the differences?

|  |  |
| --- | --- |
| **WEB SERVICE** | **WEB API** |
| **Definition:** Web service is a collection of open source protocols and standards used for exchanging data between systems or applications | **Definition:** API is a software interface that allows two applications to interact with each other without any user involvement. |
| Web service is a collection of open source protocols and standards used for exchanging data between systems. | API is a software interface that allows 2 applications to interact without any user involvement. |
| All Web services are APIs | Not all APIs are web services |
| Web service is used for **REST, SOAP and XML-RPC** for communication | API is used for any style of communication. |
| Web service supports only HTTP protocol | API supports HTTP/HTTPS protocol. URL request/Response Headers, and so on. |
| **Inside an organization, it is better because it is faster.** | **Outside de organization, it is better to use API with HTTPS.** |
| Web service supports XML  By default it is a SOAP. Api, but you can also use JSON by adding a ScriptServiceAttribute. | API supports XML and JSON |
|  |  |
| Web Service is a class derived from WebService and has automatic WSDL generation. | Web API is a Controller whose base is ApiController and does not use Views. |
| It doesn’t have lightweight design, needs a SOAP convention to send or receive data over the system. | It HAS a light-weight architecture furthermore, useful for gadgets which have constrained transmission capacity like smart phones. |
| It provides suppots |  |
| It is not open source, however, can be devoured by any customer that comprehends xml | It is an open source and also ships with .NET framework. |
| Web services can be hosted on IIS. | Web API can be hosted only on IIS and self. |
| 2,000 pages for all standards. | 2 pages of standards. |

## Advantages of API Services

Here are pros/benefits of using API:

* API supports traditional CRUD (Create Read Update Delete) actions as it works with HTTP verbs GET, PUT, POST, and DELETE.
* API helps you to expose service data to the browser
* It is based on HTTP, which is easy to define, expose in REST-full way.

## Advantages of Web Services

Here are the important pros/benefits of using web services:

* Offers faster communications within and across organizations
* Each service exists independently of other services.
* Interoperability has the highest priority.
* Using Web services, your application helps you to publish its message or function to the rest of the world.
* Web services help solve interoperability issues by giving different applications a way to link their data.
* Web services help you to exchange data between different applications and different platforms.
* It allows applications to communicate, exchange data, and shared services among themselves.
* Web services are specifically designed to be used as a web page request and help you to receive data.
* It serves as building blocks which makes it easy to reuse web service components in other services. Web Services are deployed on internet standards such as standard Apache, and Axis2. It provides WSDL, HTTP, driven services.

## Disadvantages of API

Here are important drawbacks/cons of using API services:

* Creating API is a very time-consuming process
* A fixed scale is necessary
* Imprecise boundary delineation
* To create API, programming knowledge is necessary
* Maintenance cost is very high
* It can crash when testing API

## Disadvantages of Web Services

Drawbacks/cons of using Web services:

* It does not access from browser
* Not leverage emerging Web developments (Semantic Web, AJAX XMLHttpRequest, etc.)
* Some web services are simple to use, but there are some flaws of using it.
* Any time one creates a service to handle a variety of customers, there is a demand for specialized machine requirements.
* The HTTP protocol is not reliable, so it does not offer any guarantee of delivery of the response.

**Web Services:** A **Web services** are any bit of services that makes it accessible over the Internet and normalizes its correspondence through XML encoding. A customer conjures web services by sending a solicitation (for the most part as an XML message), and the services send back an XML response. Web services summon communication over a network, with HTTP as the most widely recognized methods for the network between the two frameworks. Web services are equivalent to SOA (Services Oriented Architecture) and fundamentally depend on measures, for example, XML-RPC and SOAP (Simple Object Access Protocol).

**Components:** All the standard web services work using the following components.

* SOAP (Simple Object Access Protocol)
* UDDI (Universal Description, Discovery and Integration)
* WSDL (Web Services Description Language)

**Web APIs:** **API** stands for **Application Programming Interface**. It is a collection of communication conventions and subroutines used by various programs to communicate between them. A developer can utilize different API apparatuses to make its program simpler and less complex. Likewise, an API encourages the developers with a proficient method to build up their product programs. Thus, in simple terms, an API determines how programming segments ought to associate with one another. It is a set of protocols and schedules, and its reactions are returned as JSON or XML in data. APIs can utilize any kind of communication convention and are not restricted similarly as a web service is.

|  |  |
| --- | --- |
| Web Services - that's standard defined by W3C, so they can be accessed semi-automatically or automatically (WSDL / UDDI). The whole thing is based on XML, so anyone can call it. And every aspect of the service is very well defined. There's parameters description standard, parameter passing standard, response standard, discovery standard, etc. etc.  Despite the fact that automatic invoking and discovery is barely working because clients are rather poor, and you have no real guarantee that any service can be called from any client.  Web API is typically done as HTTP/REST, nothing is defined, output can be for eg. JSON/XML, input can be XML/JSON/or plain data. There are no standards for anything => no automatic calling and discovery. You can provide some description in text file or PDF, you can return the data in Windows-1250 instead of unicode, etc.  Web is switching towards Web API / REST. Web Services are really no better than Web API.  Very complicated to develop and they eat much more resources (bandwidth and RAM)... and because of all data conversions (REQUEST->XML->DATA->RESPONSE->XML->VALIDATION->CONVERSION->DATA) are very slow.  Eg. In WebAPI you can pack the data, send it compressed and un-compress+un-pack on the client. In SOAP you could only compress HTML request. |  |

### Types of Web Services

Web services should be implemented in various ways. The two types of widely used web services are SOAP and RESTful web services.

**SOAP** – SOAP is a protocol which was designed before REST came into the picture. The main idea behind creating SOAP was to ensure that programs built on different platforms and programming languages could securely exchange data.

**REST** – This was designed specifically for working with components such as media components, files, or even objects on a particular hardware device. Any web service which is defined on the principles of REST can be called a RESTful web service. REST uses the normal HTTP verbs of GET, POST, PUT and DELETE for working with the required components.

## Features of Web API

Here are some essential features of API:

* Efficiency
* Wider reach
* Customizable
* Personalization
* Data ownership
* Easy integration with GUI
* Time effective
* Language-independent

## Features of Web Services

Here are some essential features of web services:

* Loosely coupled
* Synchronous or asynchronous functionality
* Ability to support remote procedure calls
* Supports document exchange