Understanding Web Services

Introduction

We encountered terms like APIs and web services in the last few videos. While these terms are related to software development and integration, they have distinct meanings and purposes. This documentation will explore the some of the crucial differences between ***APIs and Web services, REST API and SOAP*** and provide simple explanations and examples to help you understand them better.

Web Services

A. What is web Service?

Web services are APIs designed to be accessed online using standard web protocols. Web services enable different software systems to communicate and exchange data regardless of the technologies or platforms they use.

B. Data Representation in web services

- Web services use XML (eXtensible Markup Language) to structure and represent data. XML allows developers to define custom markup tags and hierarchies to organise data in a structured manner. It is widely adopted and provides a human-readable format that software applications can quickly parse.

- JSON (JavaScript Object Notation) is also commonly used for data representation in web services, particularly in RESTful APIs. The choice between XML and JSON depends on specific requirements, technology stack, and developer preferences.

C. Architectural styles

There are two main architectural styles or approaches used in web services:

I. Representational State Transfer (REST)
II. Simple Object Access Protocol (SOAP)

We will explore the two architectural styles later in the document.

APIs - Application Programming Interfaces

What is API?

- API stands for ***Application Programming Interface***. In simple terms, an API is a set of rules and protocols that allows different software applications to communicate and interact with each other. It acts as a bridge between two applications, enabling them to exchange data and functionality seamlessly.

- APIs define the methods and formats to request and receive information between applications. They provide a standardised way for developers to access specific features or services offered by another application or system. Think of APIs as a messenger that helps different applications talk to each other and share information.

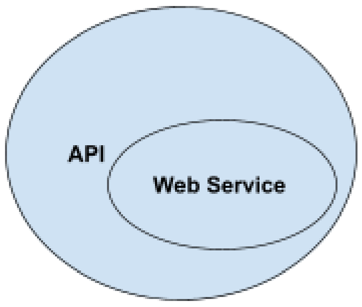
API vs Web Service via Example

Let's say you have a mobile application that displays the current weather. You must fetch weather data from a service provider to show it in your app.

- **API example:** In this scenario, you use the weather service provider's API. The API provides a set of functions or endpoints that allow you to make specific requests for weather data. You would request the API for the current weather of a particular location, and the API would respond with the relevant data in a structured format, such as JSON. You can then extract and display the weather information in your mobile app. The API acts as the intermediary, enabling your app to communicate and retrieve data from the weather service.

- **Web service example:** In this case, the weather service provider offers a web service for accessing weather information online. You would request an HTTP to the web service's specific URL and any required parameters (e.g., location). The web service would process your request and respond with the weather data in a standardised format like XML or JSON. You can then parse the response and display the weather information in your app. In this case, the web service is an API accessed via web protocols.

The critical difference lies in the underlying technologies and protocols used. An API is a general concept for facilitating communication between applications. At the same time, a web service is a specific type of API accessed over the web using web-based protocols like HTTP. Hence, all web services are API but not vice versa.



REST API vs SOAP

In web services, two popular protocols are often used for communication between applications: REST API and SOAP. These protocols define the rules and formats for exchanging data over the internet. In this article, we'll explore the differences between REST API and SOAP, using simple language and examples to help you grasp the concepts.

REST API (Representational State Transfer)

REST API is an architectural style that uses HTTP as its communication protocol. It focuses on exposing resources (such as data or functionality) as URLs, known as endpoints. These endpoints represent different operations that can be performed on the resources.

1. **Simplicity and Lightweight:**
   REST API is designed to be simple and lightweight, making it easy to understand and implement. It uses standard HTTP methods like GET, POST, PUT, and DELETE to perform actions on resources. For example, you can request an HTTP GET request to the appropriate endpoint to retrieve data from a REST API.

2. **Stateless Communication:**
   REST API follows a stateless communication model, meaning each request from a client to a server contains all the necessary information to be understood. The server doesn't keep track of the client's previous interactions. This simplifies the server implementation and allows for scalability.

3. **Data Formats:**
   REST API commonly uses JSON (JavaScript Object Notation) or XML (eXtensible Markup Language) as data formats for request and response payloads. JSON is more widely used due to its simplicity and compatibility with modern web technologies.

SOAP (Simple Object Access Protocol)

SOAP protocol uses XML to structure the data being exchanged between applications. It relies on rules for defining the message format and communication details.

1. **Complex and Robust:**
   SOAP is more complex than REST API. It provides a standardised message structure definition, including request and response formats. SOAP also offers features like encryption, security, and error handling, making it suitable for enterprise-level applications.

2. **Communication Protocol:**
   Unlike REST API, SOAP is not limited to HTTP. It can use HTTP, SMTP, or JMS (Java Message Service). This flexibility allows SOAP to be used in various communication scenarios.

3. **XML-based Messaging:**
   SOAP messages are typically formatted using XML. This structured format ensures compatibility across different platforms and programming languages. SOAP defines a specific message envelope structure, including headers and body elements.

Example - REST API vs SOAP

Let's compare REST API and SOAP based on an example of retrieving customer information from a server.

REST API:

- Endpoint: GET /customers/{id}
- Request: A GET request to the specific URL endpoint with the customer ID in the path.
- Response: The server responds with a JSON or XML representation of the customer data containing the requested information.
- Example Request: GET /customers/123
- Example Response:

```
{
  "id": 123,
  "name": "John Doe",
  "email": "johndoe@example.com",
  "address": "123 Main St"
}
```

SOAP:

- Endpoint: A SOAP message is sent to the server's SOAP endpoint.
- Request: A SOAP request message containing the customer ID as a parameter.
- Response: The server sends a SOAP response message encapsulating the requested customer information.
- Example Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://example.com/webservice">
   <soapenv:Header/>
   <soapenv:Body>
      <web:GetCustomerRequest>
         <web:CustomerId>123</web:CustomerId>
      </web:GetCustomerRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

- In this example, the REST API uses a simple HTTP GET request to retrieve customer information by specifying the customer ID in the URL. The server responds with a JSON or XML representation of the customer data.

- On the other hand, the SOAP-based approach involves constructing a SOAP request message with the customer ID as a parameter and sending it to the server's SOAP endpoint. The server then sends a SOAP response message encapsulating the requested customer information.

Overall, the REST API approach is more lightweight and uses standard HTTP protocols and data formats like JSON, while SOAP involves more complex XML-based messaging and additional protocols.

REST is favoured for simplicity and ease of use, whereas SOAP is often used in enterprise scenarios requiring more advanced features like security and transactions.

Conclusion

In conclusion, we have explored APIs, web services, and the differences between REST API and SOAP. A few of the inference we have drawn are as follows:

- APIs act as a communication bridge between software applications, while web services are APIs accessed online using web protocols. Web services commonly use XML or JSON for data representation.

- REST API follows the principles of simplicity and lightweight, using HTTP methods and JSON for data exchange. Conversely, SOAP is more complex and robust, supporting various protocols and using XML for messaging.

- The choice between REST API and SOAP depends on specific requirements. REST API is preferred for its simplicity and compatibility with modern technologies, while SOAP offers advanced features suitable for enterprise-level applications.

Understanding these concepts helps developers make informed decisions for designing and integrating software systems, ensuring efficient communication between applications.

References:

- What is the difference between API and REST API
- What are protocols?
- Request vs Response
- JSON vs XML
- REST vs SOAP
- Popular Interview question on Web Services