

# JSON, SOAP, XML VS.

	SOAP	REST
Meaning	Simple Object Access Protocol	Representational State Transfer
Design	Standardized protocol with pre-defined rules to follow.	Architectural style with loose guidelines and recommendations.
Approach	Function-driven (data available as services, e.g.: "getUser")	Data-driven (data available as resources, e.g. "user").
Statefulness	Stateless by default, but it's possible to make a SOAP API stateful.	Stateless (no server-side sessions).
Caching	API calls cannot be cached.	API calls can be cached.
Security	WS-Security with SSL support. Built-in ACID compliance.	Supports HTTPS and SSL.
Performance	Requires more bandwidth and computing power.	Requires fewer resources.
Message format	Only XML.	Plain text, HTML, XML, JSON, YAML, and others.
Transfer protocol(s)	HTTP, SMTP, UDP, and others.	Only HTTP
Recommended for	Enterprise apps, high-security apps, distributed environment, financial services, payment gateways, telecommunication services.	Public APIs for web services, mobile services, social networks.
Advantages	High security, standardized, extensibility.	Scalability, better performance, browser-friendliness, flexibility.
Disadvantages	Poorer performance, more complexity, less flexibility.	Less security, not suitable for distributed environments.

JSON is a more lightweight and less verbose format, and it's easier to read and write as well. In most cases, it's ideal for data interchange over the internet. However,

XML still has some advantages. For example, it allows you to place metadata within tags and also handles mixed content better — especially when mixed node arrays require detailed expressions.

```
<?xml version="1.0" encoding="UTF-8"?>
<authentication-context>
  <username>my_username</username>
  <password>my_password</password>
  <validation-factors>
    <validation-factor>
      <name>remote_address</name>
      <value>127.0.0.1</value>
    </validation-factor>
  </validation-factors>
</authentication-context>
```

This is how the above XML code looks in JSON:

```
{
  "username" : "my_username",
  "password" : "my_password",
  "validation-factors" : {
    "validationFactors" : [
      {
        "name" : "remote_address",
        "value" : "127.0.0.1"
      }
    ]
  }
}
```