REST
(Representational State Transfer)

RESTful Web Services

Introducción

- Nuevo enfoque para desarrollo de servicios web
- Definido por Roy Fielding, uno de los autores principales de la especificación HTTP
- Usado por Facebook, Twitter, Youtube, etc

Que es REST?

- REST interfaz entre sistemas que usa HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON

- Alternativa a SOAP (Simple Object Access Protocol) y otros protocolos para intercambio de datos mas complejos

Características REST

- Protocolo cliente/servidor sin estado
- Los objetos en REST siempre se manipulan a partir de la URI
- Interfaz uniforme
  - Operaciones bien definidas (POST, GET, PUT y DELETE)
- Sistema de capas
- Uso de hipermedios

Ventajas

- Separación entre el cliente y el servidor
- Visibilidad, fiabilidad y escalabilidad
- Independencia del tipo de plataformas o lenguajes

RESTful Web services

- El concepto central en servicios web RESTful es la noción de recursos.
- Los recursos están representados por URI.
- Los clientes envían solicitudes a estos URI utilizando los métodos definidos por el protocolo HTTP y, posiblemente, como resultado de eso, el estado del recurso afectado cambia.

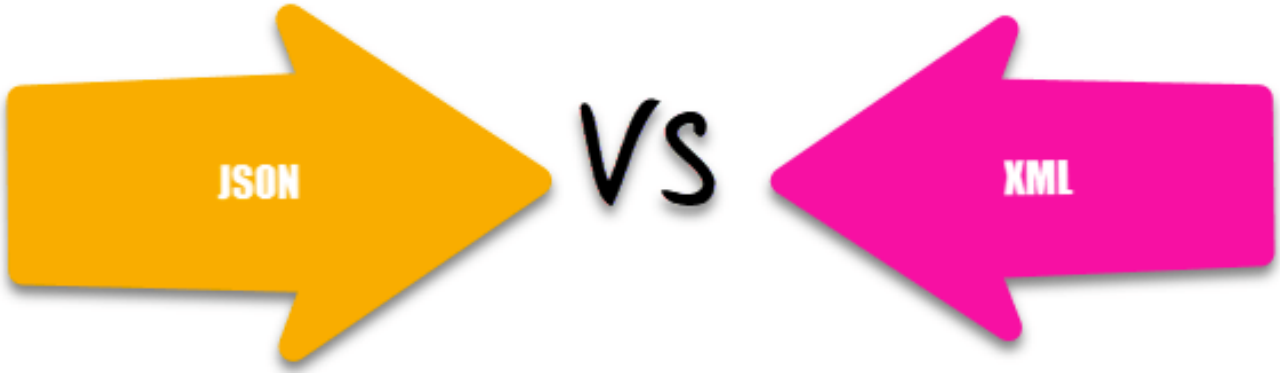
Métodos HTTP

HTTP Method	Action	Examples
GET	Obtain information about a resource	http://example.com/api/orders (retrieve order list)
GET	Obtain information about a resource	http://example.com/api/orders/123 (retrieve order #123)
POST	Create a new resource	http://example.com/api/orders (create a new order, from data provided with the request)
PUT	Update a resource	http://example.com/api/orders/123 (update order #123, from data provided with the request)
DELETE	Delete a resource	http://example.com/api/orders/123 (delete order #123)

Representación de la Información

- Los estándares para la representación de información facilitan el intercambio de información entre aplicaciones totalmente heterogéneas.

- Los mas relacionados con los web services son XML y JSON



XML (eXtensible Markup Language)

- XML es una herramienta independiente del software y el hardware, para transportar y almacenar datos.
- Las etiquetas de XML no están predefinidas como en HTML, el autor debe crear tanto las etiquetas como la estructura.
- Dada su flexibilidad puede ser usado para representar cualquier estructura de datos.

```
<catalogo>
    <libro>
        <autor>Raúl González Duque</autor>
        <titulo>Python para todos</titulo>
        <genero>Computación</genero>
    </libro>
</catalogo>
```

JSON (JavaScript Object Notation)

- JSON es un subconjunto de JavaScript aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente.

- Es un estándar abierto que utiliza texto plano para codificar información en la forma atributo: valor.

```
{
  libro: {
    autor: Raúl González Duque,
    titulo: Python para todos,
    genero: Computación
  }
}
```

XML vs. JSON

JSON	XML
JSON object has a type	XML data is typeless
JSON types: string, number, array, Boolean	All XML data should be string
Data is readily accessible as JSON objects	XML data needs to be parsed.
JSON is supported by most browsers.	Cross-browser XML parsing can be tricky
JSON has no display capabilities.	XML offers the capability to display data because it is a markup language.
JSON supports only text and number data type.	XML support various data types such as number, text, images, charts, graphs, etc. It also provides options for transferring the structure or format of the data with actual data.
Retrieving value is easy	Retrieving value is difficult

XML vs. JSON

A fully automated way of deserializing/serializing JavaScript.	Developers have to write JavaScript code to serialize/de-serialize from XML
Native support for object.	The object has to be express by conventions - mostly missed use of attributes and elements.
It supports only UTF-8 encoding.	It supports various encoding.
It doesn't support comments.	It supports comments.
JSON files are easy to read as compared to XML.	XML documents are relatively more difficult to read and interpret.
It does not provide any support for namespaces.	It supports namespaces.
It is less secured.	It is more secure than JSON.

Construcción de un API Rest en Python Flask

- Ver Practica en Classroom

Referencias

- API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos
- Designing a RESTful API with Python and Flask https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask
- JSON vs XML: What's the Difference? https://www.guru99.com/json-vs-xml-difference.html