Laboratorio I

Introducción

En el mundo de las aplicaciones web conforme el crecimiento exponencial de las plataformas de consumo de datos siempre ha surgido la necesidad de almacenar y transportar cada vez más datos a través de internet. Los APIs se han convertido en el corazón de todo proyecto informático, permitiendo el acceso intermediario a datos de un servidor o una base de datos.
Los EndPoint nos permiten la comunicación entre un servidor desde una aplicación, algo así como el portero de un hotel, el cual nos puede dirigir y al mismo tiempo impedirnos el ingreso a este.
La comunicación entre un API y una aplicación a través de un EndPoint es limitada a texto, por lo que es necesario otorgar un formato a nuestros datos para poder identificar lo que necesitamos saber al tratar de procesar estos datos, existen varias soluciones pero las más comunes para el tratamiento de datos son JSON y XML.

JSON (JavaScript Object Notation)

¿Qué es?

Es una solución basada en JavaScript que nos presenta una sintaxis la cual permite procesar y serializar todo objeto de JavaScript.

Aun cuando JSON está basado en JavaScript, no todo objeto de JavaScript se puede convertir a JSON y no todo JSON se puede convertir en un objeto de JavaScript. Se puede encontrar más información al respecto en este link.

¿Para qué sirve?

JSON nace para familiar la comunicación entre cliente/servidor, debido a su simple estructura este permite una fácil y rápida comunicación e interpretación.

¿Tipos de datos?

String

Permite el tratamiento y almacenamiento de todo valor textual incluyendo valores UNICODE y caracteres de control (\", \\, \/, \f, \n, \b, \r, \t, \u)

Estructura

Los tipos de dato String se conforman de la siguiente forma

```
{
     “propertyName”: [“ + Texto + ”],
     "nombre": "John”,

     “propertyName”: [“ + Texto + \b + Texto + “]
     "biografia": "Estudiado de Harvard\bProveniente de Chicago Illinois"
}
```

Number

Permite el tratamiento y almacenamiento de todo valor numeral incluyendo todo numero racional y operación matemática simple

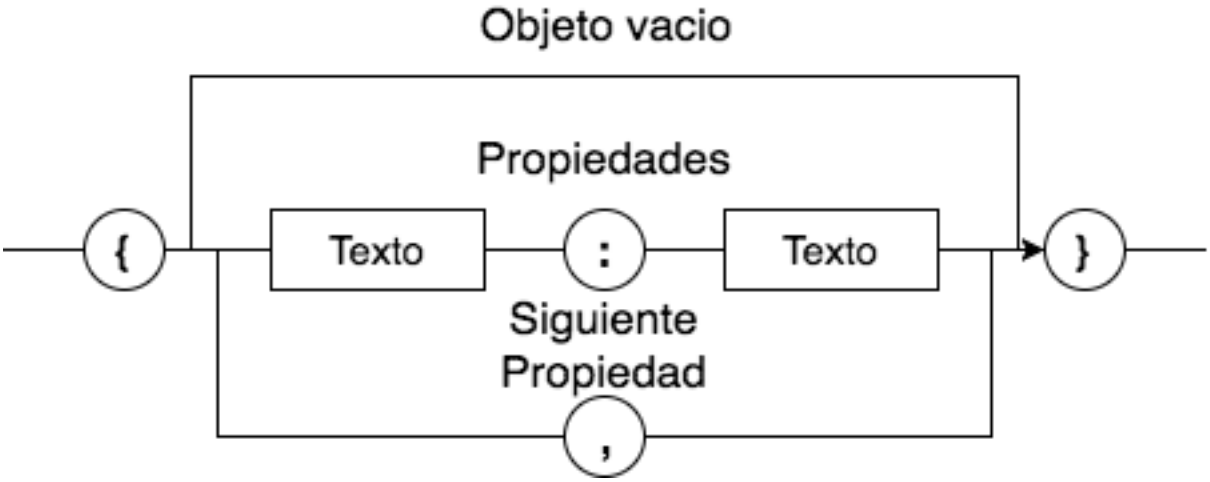
Estructura

```
{
     “propertyName”: Numero + . + Número, (Decimal)
     "altura": 1.83,

     “propertyName”: Numero,
     "edad": 21,

     “propertyName”: Operación Matemática (Resultado),
     "operacion": (-10 * 5 + 10) / 20 // -2
               (La operación es procesada y se almacena el resultado)
}
```

Object

Un objeto de Json está conformado por propiedades de diferentes tipos de dato.
Los objetos pueden ser anidados, por lo que pueden contener objetos dentro de sus propiedades

Estructura


Objeto vacio
Propiedades
{
Texto
:
Texto
}
Siguiente
Propiedad
,

```
Objeto Vacío : {}
Objeto con una propiedad :
{
     “propertyName”: Value
}
Objeto con múltiples propiedades :
{
     “propertyName”: Value, (Multiples propiedades se separan por ,)
     “propertyName2”: Value2
}
Objeto anidado :
{
     “propertyName”: { (Objeto anidado)
        “propertyName”: Value
     }
}
```

Array

Una estructura de tipo objeto cuyo constructor le permite tomar la forma de colección de datos. Los arreglos de JavaScript y JSON a diferencia de los lenguajes de programación tradicionales, poseen la posibilidad de que sus valores sean de diferentes tipos de dato. Los arreglos al igual que los objetos, pueden ser anidados.

Estructura

```
Arreglo Vacio : []
Arreglo de un tipo :
{
     “arrayName”: [
        “text”,
        “text”
     ] (Un tipo)
}
Arreglo de múltiples tipos :
{
     “arrayName”: [
        “text”,
        123,
        null,
        undefined
     ]
}
Arreglo anidado :
{
     “arrayName”: [
        [
           4,
           “text”
        ]
     ]
}
```

Boolean

El tipo booleano sigue las mismas reglas que JavaScript, sus unicos valores son TRUE y FALSE

Estructura

```
{
     “equipoPersonal”: true,
     “dispositivoMovil”: false,
}
```

Null

El valor NULL define la ausencia intencional del valor de un objeto.

Estructura

```
{
     “objetivos”: null
}
```

Undefined

El valor Undefined define indefinición de un objeto o valor de JavaScript

NaN

El valor Undefined define invalidación de un numero en JavaScript

REST

Entre más grande se vuelve una aplicación, más difícil es controlar su escalabilidad. Más usuarios alrededor del mundo requieren accesos multiplataforma, rápidos y confiables a nuestros datos. Es imposible componer una interface de conexión entre cada plataforma hacia nuestros servidores.

¿Qué es?

REST es una interface cliente/servidor basada en el protocolo HTTP. Esto nos permite interactuar desde cualquier lenguaje o plataforma a través de los lenguajes de intercambio de información JSON o XML. REST pone a disposición una serie de EndPoints en nuestro servidor al cual se le pueden realizar transacciones.

Podríamos afirmar que REST funciona igual a un sitio web, una solicitud sale de nuestro browser hacia un servidor y recibimos un HTML con el sitio a mostrar, solo que REST permite recibir JSON y XML.

¿Cómo funciona?

REST como fue mencionado anteriormente, ofrece una serie de EndPoints accesibles en nuestro sitio.

Estos EndPoints pueden ser consultados atraves de una serie de métodos HTTP, estos permitirán ejecutar un query en nuestros servidores y retornar información en formato JSON o XML.

Los request realizados a un REST API están conformados por Request y Reponse

Request

Contiene toda la información necesaria para que el servidor procese y retorne los datos solicitados. Los objetos o propiedades más comunes contenidos en un request son

Header:

Contiene toda la información de autentificación y configuración del protocolo de HTTP.

Body:

Contiene todos los parámetros enviados a un EndPoint, estos pueden venir en forma de form, archivo binario o raw-text que va desde texto plano hasta JSON y XML

URL:

Contiene la URL y con ella todo query param enviado a través del request.

Method:

Método a ejecutar por el servidor dada la solicitud del cliente

Métodos REST (CRUD)

Para cada parte del CRUD existe un equivalente en REST para las diferentes solicitudes a un servidor, esto permite la reinterpretación de EndPoints para distintos propósitos.

POST (Create)

Es el método utilizado por REST para en envió de información datos para la creación de objetos.

GET (Read)

Es el método utilizado por REST para la lectura de información al servidor, por su naturaleza y propósito, carece de BODY.

PUT (Update)

Es el método utilizado por REST para la actualización de datos en el servidor.

DELETE (Delete)

Es el método utilizado por REST para la eliminación de datos en el servidor

WCF

¿Qué es?

Es un conjunto de librería creadas por Microsoft basadas en el Framework ASP.NET, su utilidad principal es crear aplicaciones orientadas a servicios que se encargan de la comunicación entre 2 servidores mediante mensajes asincrónicos.

¿Cómo se utiliza?

Utiliza un protocolo de llamados SOAP el cual se encarga de construir el paquete de solicitud al servidor incluyendo los argumentos y resultados esperados del lado del cliente. SOAP está basado en XML y ofrece una variedad de métodos los cuales son ampliamente configurables y retro compatibles con todas las tecnologías de Microsoft

Bibliografía

Andreas Grech. (Jan 23, 2010). What is JSON and why would I use it?. Retrieved January 18, 2018, from https://stackoverflow.com/questions/383692/what-is-json-and-why-would-i-use-it

BBVA. (n.d.). API Rest. Retrieved January 18, 2018, from https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos

MDN. (n.d.). JSON. Retrieved January 18, 2018, from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON

MDN. (n.d.). Null. Retrieved January 18, 2018, from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/nullThe timeless

MDN. (n.d.). Null. Request January 18, 2018, from https://developer.mozilla.org/es/docs/Web/API/Request

timeless repository. (n.d.). Retrieved January 18, 2018, from http://timelessrepo.com/json-isnt-a-javascript-subset

Paul Osman. (Jan 23, 2010). What is an Endpoint?. Retrieved January 18, 2018, from https://stackoverflow.com/questions/2122604/what-is-an-endpoint

W3. (n.d.). JSON. Retrieved January 18, 2018, from https://www.w3schools.com/js/js_json_intro.asp

WebConcepts (July 14, 2014) REST API concepts and examples. Retrieved January 12, 2018, from https://www.youtube.com/watch?v=7YcW25PHnAA