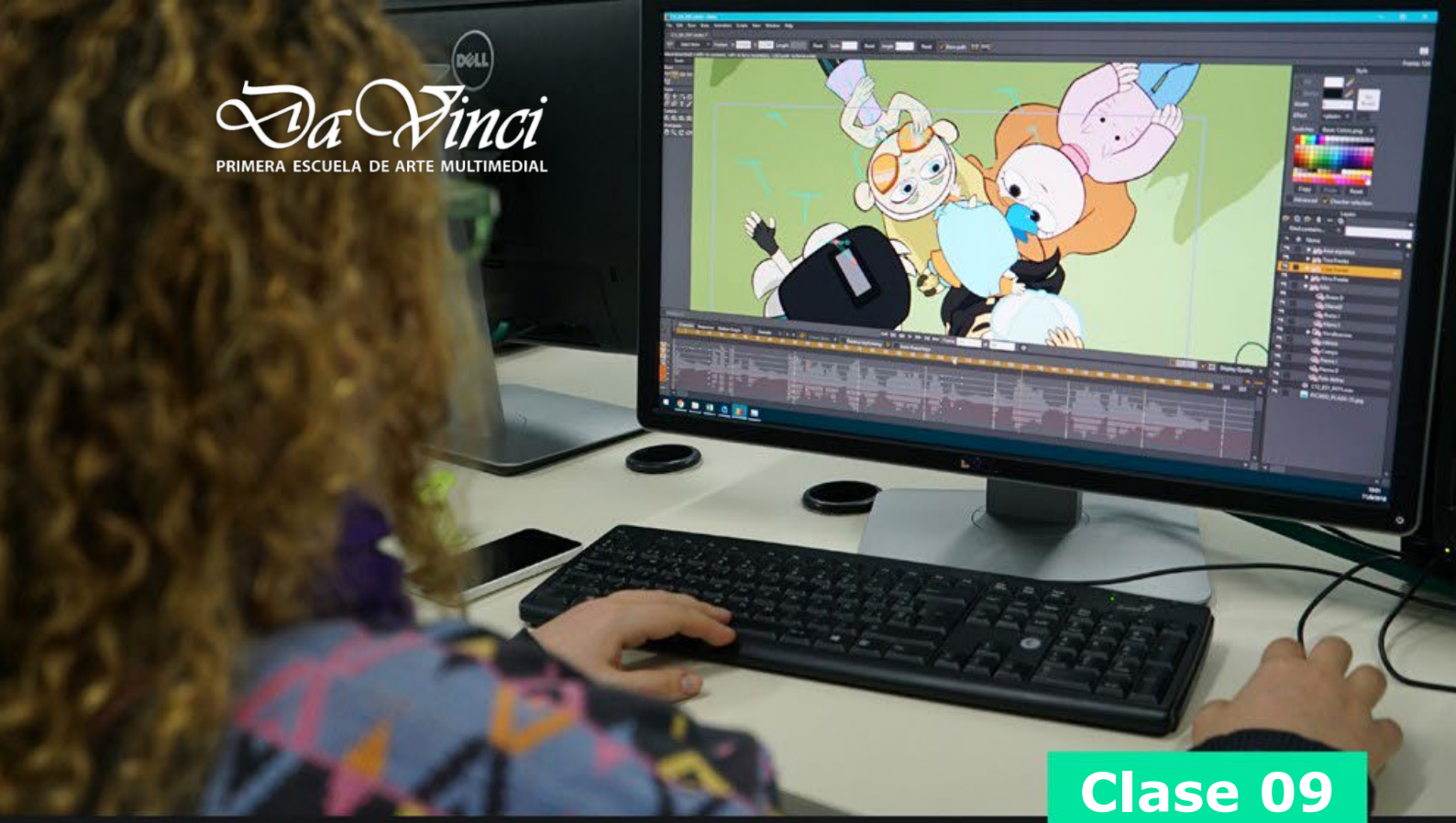


Clase 09

# Análisis de Sistemas

## Materia:
Programación Web II

**Docente contenidista:** ROLDÁN, Hernán

**Revisión:** Coordinación

# Contenido


Introducción a JSON ..... 3

¿Qué es JSON? ..... 3
Estructura y sintaxis de JSON ..... 3
Objetos: ..... 3
Arreglos: ..... 3

Diferencias entre JSON y otros formatos de intercambio de datos ..... 4

Librería comúnmente usada en PHP ..... 5

Codificación y Decodificación JSON en PHP ..... 5

Función json_encode() ..... 5
Función json_decode() ..... 6

Parámetros: ..... 6

a) Acceso a propiedades en objetos JSON ..... 8
b) Modificación de objetos JSON ..... 9
c) Agregar nuevos elementos a un objeto JSON ..... 10

Consumir datos JSON ..... 11

Bibliografía ..... 14

Para ampliar la información ..... 14

# Clase 9

¡Te damos la bienvenida a la materia **Programación Web II**!

**En esta clase vamos a ver los siguientes temas:**

- Introducción a JSON.
- Librería comúnmente usada en PHP.
- Codificación y Decodificación JSON en PHP.
- Trabajo con objetos JSON.
- Consumir datos JSON.

# Introducción a JSON

## ¿Qué es JSON?

JSON (JavaScript Object Notation) es un formato de intercambio de datos ligero y muy utilizado en el desarrollo web y aplicaciones en general. Fue popularizado por su similitud con la notación de objetos en JavaScript, pero JSON es independiente de lenguajes de programación y ampliamente soportado.

JSON se utiliza para representar y transmitir datos estructurados entre diferentes sistemas. Es legible para los humanos y fácilmente interpretable por las computadoras, lo que lo hace muy adecuado para el intercambio de datos en aplicaciones web y servicios web (APIs).

## Estructura y sintaxis de JSON

JSON se basa en dos estructuras de datos principales: objetos y arreglos. Ambos están representados de forma anidada, lo que permite la creación de estructuras complejas.

## Objetos:

Un objeto JSON es un conjunto no ordenado de pares clave-valor. Las claves (nombres de propiedad) deben estar entre comillas dobles, seguidas de dos puntos, y los valores pueden ser cualquier valor JSON válido (cadena, número, objeto, arreglo, booleano o null). Los pares clave-valor se separan por comas, y el objeto completo se encierra entre llaves {}.

*Ejemplo de objeto JSON:*

```
{
  "nombre": "Juan",
  "edad": 30,
  "ciudad": "Madrid"
}
```

## Arreglos:

Un arreglo JSON es una lista ordenada de valores, separados por comas y encerrados entre corchetes [].

*Ejemplo de arreglo JSON:*

```
["manzana", "naranja", "plátano"]
```

# Diferencias entre JSON y otros formatos de intercambio de datos

Algunas de las diferencias más relevantes entre JSON y otros formatos de intercambio de datos, como XML y CSV, son:

- **Legibilidad**: JSON es más legible para los humanos en comparación con XML y CSV, lo que facilita su lectura y depuración.
- **Tamaño del archivo**: JSON tiende a ser más compacto que XML debido a su sintaxis más sencilla, lo que resulta en tamaños de archivo más pequeños y menor uso de ancho de banda.
- **Facilidad de uso**: La sintaxis de JSON se asemeja mucho a la de los objetos en muchos lenguajes de programación, lo que lo hace más fácil de manejar y manipular en comparación con XML o CSV.
- **Estructura de datos**: JSON es más adecuado para representar datos estructurados en forma de objetos y arreglos, mientras que XML es más versátil y permite representar estructuras de datos más complejas.
- **Soporte del navegador**: JSON es soportado nativamente por JavaScript, lo que lo hace una elección natural para el intercambio de datos entre aplicaciones web cliente y servidores.

*En resumen, JSON es un formato ligero y eficiente para el intercambio de datos en aplicaciones web y servicios web. Su simplicidad y legibilidad lo convierten en una elección popular para transmitir datos estructurados entre sistemas.*

# Librería comúnmente usada en PHP

La librería más comúnmente utilizada para trabajar con JSON en PHP es la función incorporada json_encode() y json_decode().

Estas funciones están disponibles en versiones de PHP 5 y posteriores, por lo que son ampliamente compatibles y fáciles de usar.

# Codificación y Decodificación JSON en PHP

## Función json_encode()

La función **json_encode()** en PHP se utiliza para convertir datos PHP en formato JSON.

Podemos pasarle una variable que contenga una estructura de datos PHP, como un arreglo o un objeto, y la función generará su representación equivalente en formato JSON.

**Parámetros:**

- **$data:** Es el dato PHP que deseamos convertir a JSON. Puede ser un arreglo, un objeto, una cadena, un número, un booleano o null.

- **$options (opcional):** Es un parámetro entero que proporciona opciones adicionales para la codificación JSON. Podemos utilizar constantes como JSON_PRETTY_PRINT para obtener una salida con formato más legible.

- **$depth (opcional):** Es la profundidad máxima de anidamiento permitida para codificar el JSON. Si nuestros datos son altamente anidados, podemos ajustar este valor según nuestras necesidades.

*Ejemplo de uso de la función json_encode():*

```php
<?php

  $data = array("nombre" => "Juan", "edad" => 30, "ciudad" => "Madrid");
  $jsonData = json_encode($data);

  echo $jsonData;
  // Salida: {"nombre":"Juan","edad":30,"ciudad":"Madrid"}

?>
```

# Función json_decode()

La función json_decode() en PHP se utiliza para convertir una cadena de texto en formato JSON en una estructura de datos PHP. Esto es útil cuando necesitamos procesar datos JSON recibidos de una API externa o almacenados en un archivo.

**Parámetros:**

- **$json:** Es la cadena JSON que deseamos decodificar en datos PHP.
- **$assoc (opcional):** Si se establece en true, los objetos JSON se convertirán en arreglos asociativos en lugar de objetos PHP.
- **$depth (opcional):** Es la profundidad máxima de anidamiento permitida para decodificar el JSON.
- **$options (opcional):** Opciones adicionales para la decodificación JSON.

*Ejemplo de uso de la función json_decode():*

```php
<?php

  $jsonString =
'{"nombre":"María","edad":25,"ciudad":"Barcelona"}';
  $phpData = json_decode($jsonString, true);

  var_dump($phpData);

  /*
  Salida:

  array(3) {
    ["nombre"]=>
    string(5) "María"
    ["edad"]=>
    int(25)
    ["ciudad"]=>
    string(9) "Barcelona"
  }

  */

?>
```

En el ejemplo anterior, **json_decode()** convierte el JSON en un arreglo asociativo de PHP, lo que permite acceder a los datos como cualquier otro arreglo PHP

*Es importante tener en cuenta que estas funciones pueden manejar otros tipos de datos en PHP, como objetos y valores escalares (cadenas, números, booleanos y null), además de arreglos. La combinación de json_encode() y json_decode() permite intercambiar datos estructurados fácilmente entre PHP y otros sistemas que utilicen JSON como formato de intercambio.*

# Trabajo con objetos JSON

En PHP, después de decodificar una cadena JSON, el resultado es un objeto estándar de PHP, o un arreglo asociativo si así se indica.

Esto hace que trabajar con objetos JSON sea bastante sencillo y familiar para quienes están acostumbrados a trabajar con objetos en PHP.

## a) Acceso a propiedades en objetos JSON

Para acceder a las propiedades en un objeto JSON, se utilizan los operadores de objeto (->). Si el JSON se decodifica como un objeto PHP, puedes acceder a sus propiedades directamente utilizando el nombre de la propiedad como clave.

*Ejemplo de acceso a propiedades de un objeto JSON:*

```php
<?php

  $jsonObj = '{"nombre":"Juan","edad":30,"ciudad":"Madrid"}';
  $obj = json_decode($jsonObj);

  echo "Nombre: " . $obj->nombre . "<br>";
  echo "Edad: " . $obj->edad . "<br>";
  echo "Ciudad: " . $obj->ciudad . "<br>";

?>
```

## b) Modificación de objetos JSON

Podemos modificar las propiedades de un objeto JSON como lo haríamos con cualquier objeto PHP. Simplemente accedemos a la propiedad utilizando el operador de objeto (->) y asignamos un nuevo valor a esa propiedad.

*Ejemplo de modificación de un objeto JSON:*

```php
<?php


  $jsonObj = '{"producto":"Laptop","precio":800,"disponible":true}';

  $obj = json_decode($jsonObj);


  $obj->precio = 900; // Modificar el precio


  // Codificar el objeto modificado nuevamente a JSON

  $jsonData = json_encode($obj);


  echo $jsonData;

  // Salida: {"producto":"Laptop","precio":900,"disponible":true}


?>
```

## c) Agregar nuevos elementos a un objeto JSON

Para agregar un nuevo elemento (propiedad) a un objeto JSON, simplemente asignamos un nuevo valor a una clave que no exista previamente.

*Ejemplo de agregación en un objeto JSON:*

```php
<?php

  $jsonObj = '{"nombre":"Juan","edad":30}';
  $obj = json_decode($jsonObj);

  $obj->ciudad = "Madrid"; // Agregar nueva propiedad 'ciudad'

  // Codificar el objeto modificado nuevamente a JSON
  $jsonData = json_encode($obj);

  echo $jsonData;
  // Salida: {"nombre":"Juan","edad":30,"ciudad":"Madrid"}

?>
```

En el ejemplo anterior, hemos agregado la nueva propiedad "ciudad" al objeto JSON y, luego, hemos codificado el objeto nuevamente a formato JSON.

*Recordá que si modificás un objeto PHP después de decodificar un JSON, al codificarlo nuevamente, obtendrás el JSON actualizado con las modificaciones realizadas. Esto te permite manipular los datos JSON con facilidad y flexibilidad en tu aplicación PHP.*

# Consumir datos JSON

La API de BlueLytics proporciona datos sobre el tipo de cambio oficial y paralelo en Argentina.

En este ejemplo, obtendremos los datos del tipo de cambio oficial más reciente desde la API.

*Ejemplo de cómo consumir datos de una API:*

```php
<?php

  // URL de la API de BlueLytics para obtener el tipo de cambio
oficial más reciente
  $apiUrl = "https://api.bluelytics.com.ar/v2/latest";

  // Realizar la solicitud HTTP a la API de BlueLytics y obtener la
respuesta
  $response = file_get_contents($apiUrl);

  // Decodificar la respuesta JSON en un arreglo asociativo
  $data = json_decode($response, true);

  // Verificar si se obtuvieron los datos correctamente
  if(isset($data['oficial'])){
    // Obtener el tipo de cambio oficial
    $dolar_oficial_compra = $data['oficial']['value_buy'];
    $dolar_oficial_venta = $data['oficial']['value_sell'];
    // Obtener el tipo de cambio blue
    $dolar_blue_compra = $data['blue']['value_buy'];
    $dolar_blue_venta = $data['blue']['value_sell'];

    // Mostrar el tipo de cambio oficial
    echo "Dólar Oficial" . "\n";
    echo "-------------" . "\n";
    echo "Tipo de cambio oficial promedio para la compra: ARS " .
$dolar_oficial_compra . "\n";
    echo "Tipo de cambio oficial promedio para la venta: ARS " .
$dolar_oficial_venta . "\n";
    echo "\n";
    // Mostrar el tipo de cambio blue
    echo "Dólar Blue" . "\n";
    echo "----------" . "\n";
    echo "Tipo de cambio blue promedio para la compra: ARS " .
$dolar_blue compra . "\n";
    echo "Tipo de cambio blue promedio para la venta: ARS " .
$dolar_blue_venta . "\n";
  }
  else{
    echo "No se pudieron obtener los datos del tipo de cambio
oficial.";
  }

?>
```

En este ejemplo, hacemos una solicitud HTTP a la API de BlueLytics para obtener los datos del tipo de cambio oficial más reciente. Utilizamos ***file_get_contents()*** para obtener la respuesta de la API.

Luego, decodificamos la respuesta JSON en un arreglo asociativo utilizando ***json_decode()***. Si se obtuvieron los datos correctamente, extraemos el tipo de cambio oficial del arreglo y lo mostramos en la pantalla.

*Recordá que, al consumir datos de APIs externas, es importante verificar si la API tiene límites de uso o si requiere autenticación.*

Hemos llegado así al final de esta clase en la que vimos:

- Introducción a JSON.
- Introducción a JSON.
- Librería comúnmente usada en PHP.
- Codificación y Decodificación JSON en PHP.
- Trabajo con objetos JSON.

  Consumir datos JSON.

Te esperamos en la **clase en vivo** de esta semana.
No olvides realizar el **desafío semanal**.

**¡Hasta la próxima clase!**

# Bibliografía


Documentación Oficial de PHP

W3Schools: PHP Tutorial

Cabezas Granado, Luis. González Lozano, F. (2018) Desarrollo web con PHP y MySQL. Anaya Multimedia.

Heurtel, O. (2016) Desarrolle un sitio web dinámico e interactivo. Eni Ediciones.

Welling,Luke Thompson, L. (2017) Desarrollo Web con PHP y MySQL. Quinta Edición (2017, Editorial Anaya),


# Para ampliar la información

PHP: Funciones de JSON

PHP: The Right Way

Todo sobre PHP

PHP Programming Language Tutorial - Full Course

PHP Full Course for non-haters

CURSO de php desde cero

MDN: Introducción al lado Servidor

Guía de HTML