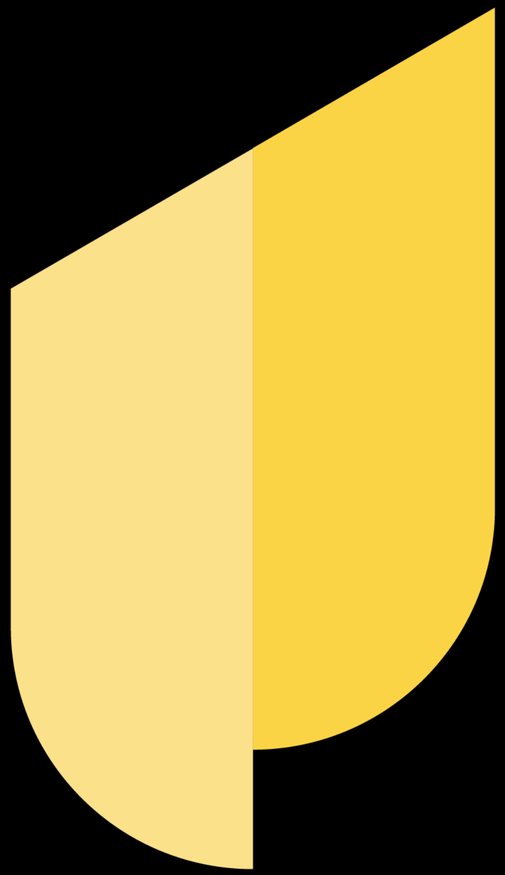
UNIVERSIDAD INTERNACIONAL DE LAS AMÉRICAS

Ingeniería en Informática / Ingeniería en Sistemas de Información / Ingeniería de Software

Programación III

Lic. Rolando Sequeira Víctor

JavaScript



JSON

¿Qué es?

- (JavaScript Object Notation es un formato sencillo para el intercambio de datos.
- El formato JSON permite representar estructuras de datos (arreglos) y objetos (arreglos asociativos) en forma de texto.
- La notación de objetos mediante JSON es una de las características principales de JavaScript y es un mecanismo definido en los fundamentos básicos del lenguaje.

- En los últimos años, JSON se ha convertido en una alternativa al formato XML, ya que es más fácil de leer y escribir, además de ser mucho más conciso.

- No obstante, XML es superior técnicamente porque es un lenguaje de marcado, mientras que JSON es simplemente un formato para intercambiar datos.

Características

- JSON es solo un formato de datos.
- Requiere usar comillas dobles para las cadenas y los nombres de propiedades. Las comillas simples no son válidas.
- Una coma o dos puntos mal ubicados pueden producir que un archivo JSON no funcione.
- Puede tomar la forma de cualquier tipo de datos que sea válido para ser incluido en un JSON, no solo arreglos u objetos.
- A diferencia del código JavaScript, en el que las propiedades del objeto pueden no estar entre comillas, en JSON solo las cadenas entre comillas pueden ser utilizadas como propiedades.

Ventajas

- Es autodescriptivo y fácil de entender.
- Su sencillez le ha permitido posicionarse como alternativa a XML.
- Es más rápido en cualquier navegador.
- Es más fácil de leer que XML.
- Es más ligero (bytes) en las transmisiones.
- Se parsea más rápido.
- Velocidad de procesamiento alta.
- Puede ser entendido de forma nativa por los analizadores de JavaScript.

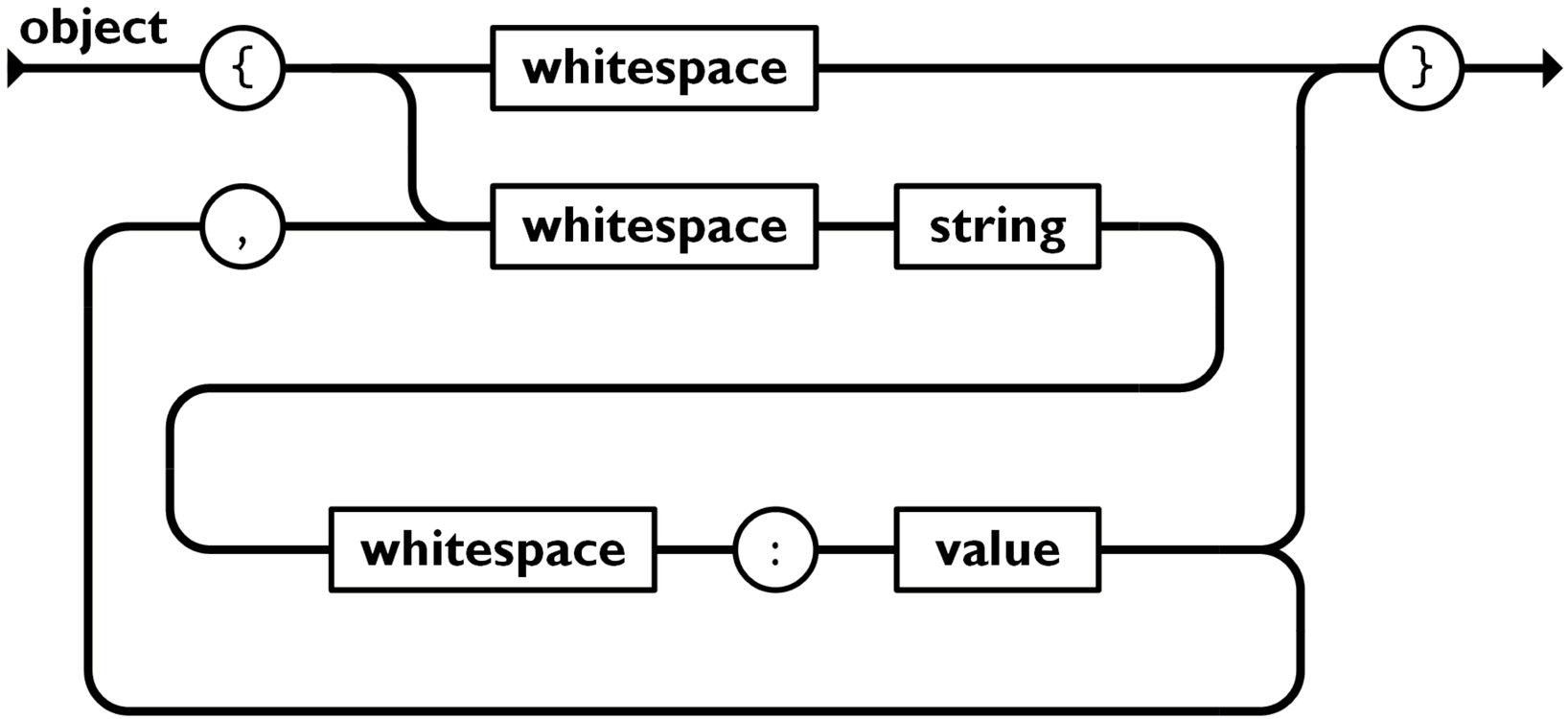
Desventajas

- Algunos desarrolladores encuentran su escueta notación algo confusa.
- No cuenta con una característica que posee XML: extensibilidad.
- No soporta grandes cargas, solo datos comunes.
- Para la seguridad requiere de mecanismos externos como expresiones regulares.

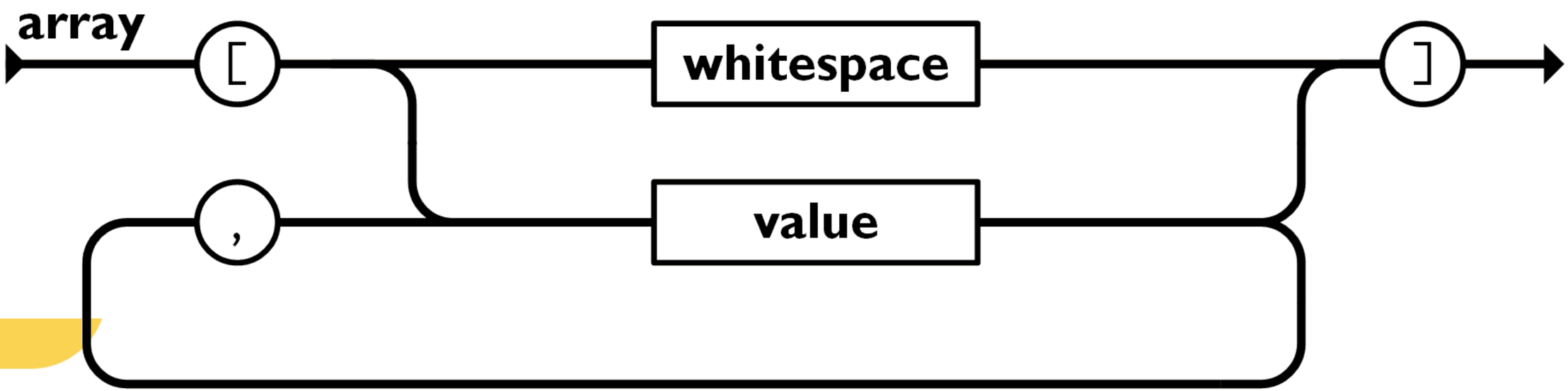
Sintaxis JSON

- JSON se basa en dos estructuras:
  - Una colección de pares de nombre/valor. En varios idiomas, esto se realiza como un objeto, registro, estructura, diccionario, tabla hash, lista con clave o matriz asociativa.
  - Una lista ordenada de valores. En la mayoría de los lenguajes, esto se realiza como una matriz, un vector, una lista o una secuencia.
- Un objeto JSON es un conjunto desordenado de pares nombre/valor. Un objeto comienza con { llave izquierda y termina con } llave derecha. Cada nombre va seguido de : dos puntos y los pares de nombre/valor están separados por ,coma.

Sintaxis JSON



Sintaxis arreglo de objeto JSON



Ejemplos

```
{"nombre": "Pepito Conejo", "edad": 25, "carnet de conducir": true}
```

```
[
  {
    "nombre": "Pepito Conejo",
    "edad": 25,
    "carnet de conducir": true
  },
  {
    "nombre": "Ana Barberá",
    "edad": 90,
    "carnet de conducir": false
  }
]
```

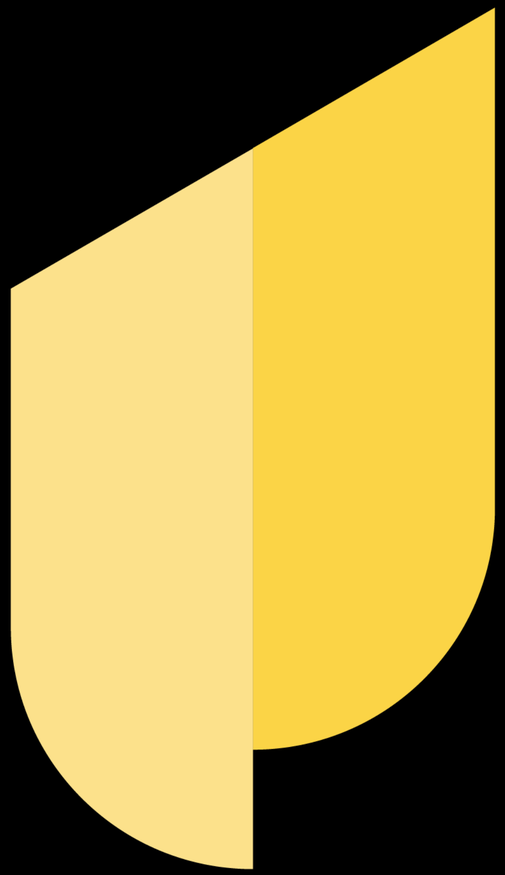
Validadores JSON

- https://jsonformatter.curiousconcept.com/
- https://jsonlint.com/
- https://codebeautify.org/jsonvalidator

Documentación completa

- https://www.json.org/json-en.html

JavaScript



Jquery

¿Qué es jQuery?

- jQuery es una biblioteca de JavaScript rápida, pequeña y rica en funciones.

- Hace que procesos como el recorrido y la manipulación de documentos HTML, el manejo de eventos, la animación y Ajax sean mucho más simples con una API fácil de usar que funciona en una multitud de navegadores.

- Documentación oficial

- jQuery Learning Center

Selectores jQuery

- El concepto más básico de jQuery es el de "seleccionar algunos elementos y realizar acciones con ellos".
- Se basa en selectores CSS
- Sintaxis selector jQuery
  - $(“selectorCSS”)

Ejemplos de selectores

- Selección de elementos en base a su id:
  - $('#Id');
- Selección de elementos en base al nombre de clase
  - $('.clase);
  - $(‘div .clase);
- Selección de elementos por su atributo
  - $('input[name=first_name]');
- Selección de elementos en forma de selector CSS
  - $('#contenedor ul.people li');

Recorrer el DOM con jQuery

- // seleccionar el inmediato y próximo elemento <p> con respecto a H1
- $('h1').next('p');
- // seleccionar el elemento contenedor a un div visible
- $('div:visible').parent();
- // seleccionar el elemento <form> más cercano a un input
- $('input[name=first_name]').closest('form');
- // seleccionar todos los elementos hijos de #myList
- $('#myList').children();
- // seleccionar todos los items hermanos del elemento
- $('li.selected').siblings();

CSS con Jquery

- JQuery incluye una manera útil de obtener y establecer propiedades CSS a los elementos.
- Las propiedades CSS que incluyen como separador un guion del medio, en JavaScript deben ser transformadas a su estilo CamelCase. Por ejemplo, cuando se la utiliza como propiedad de un método, el estilo CSS font-size deberá ser expresado como fontSize.
- Sin embargo, esta regla no es aplicada cuando se pasa el nombre de la propiedad CSS al método $.fn.css — en este caso, los dos formatos (en CamelCase o con el guión del medio) funcionarán.

Obtener valor de propiedades CSS

- $('h1').css('fontSize'); // devuelve una cadena de caracteres como "19px"
- $('h1').css('font-size'); // también funciona

Modificar propiedades CSS

- // establece una propiedad individual CSS
- $('h1').css('fontSize', '100px');
- // establece múltiples propiedades CSS
- $('h1').css({ 'fontSize' : '100px', 'color' : 'red' });

Manipulando Clases con jQuery

- var $h1 = $('h1');
- $h1.addClass('big');
- $h1.removeClass('big');
- $h1.toggleClass('big');
- if ($h1.hasClass('big')) { ... }

Modificando atributos con jQuery

- Los atributos de los elementos HTML que conforman una aplicación pueden contener información útil, por eso es importante poder establecer y obtener esa información.
- El método $.fn.attr actúa tanto como método establecedor como método GET.
- Además, al igual que el método $.fn.css, cuando se lo utiliza como método SET, puede aceptar un conjunto de palabra clave-valor o un objeto conteniendo más conjuntos.
- •$.fn.attr Obtiene o establece el valor de un determinado atributo.

Establecer atributos

- $('a').attr('href', 'contacto.html');
- $('a').attr({
- 'title' : 'Ir a la Pagina de Información',
- 'href' : info.html'
- });

Obtener atributos

- $('a').attr('href');

Modificar o obtener el contenido de un nodo

- $.fn.html Obtiene o establece el contenido HTML de un elemento.

Modificar o obtener el contenido de un campo formulario

- $.fn.val Obtiene o establece el valor (value) en elementos de formularios.

Delegación de eventos

- Delegación de eventos es un mecanismo a través del cual se evita asignar eventos a múltiples nodos específicos del DOM, asignando un event listener a solo **un nodo padre** que contiene el resto de estos nodos.

- La gran ventaja de usar delegación de eventos es que por ejemplo, de tener x elementos con la clase botón, solo se debe registrar un event listener para todos estos elementos, mientras que sin delegación de eventos se registraría un event listeners por cada nodo.

- Otra ventaja que ofrece utilizar delegación de eventos es si luego se agregan elementos dinámicamente con la clase botón, no habrá un event listener para este elemento, y la delegación nos permite solucionar ese problema.

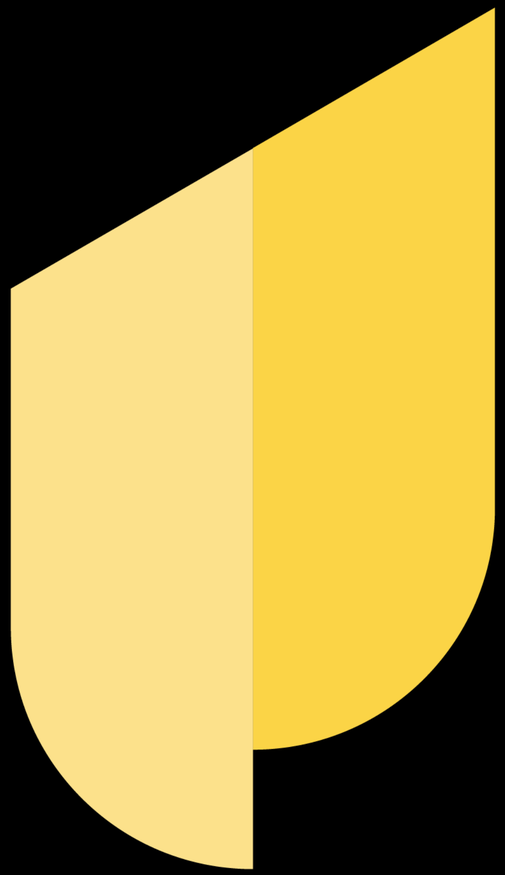
Sintaxis delegación evento JS

- nodoPadre.addEventListener(event, function);

Sintaxis delegación evento jQuery

- $('selector').on('evento', selector, datos, función);

jQuery



Animaciones

- Jquery posee los eventos $.fn.hide() y $.fn.show() para ocultar y mostrar elementos del DOM, la sintaxis de ambas funciones son las siguientes:
  - $(selector).hide(speed,callback);
  - $(selector).show(speed,callback);
  - $(selector).toggle(speed,callback);tor).toggle(speed,callback);

- Además de los métodos hide() y show() , existen otros métodos para animar los elementos de una página con jQuery: fadeIn(speed,callback) , fadeOut(speed,callback) , fadeTo(speed,callback) , slideDown(speed,callback) , slideUp(speed,callback) , slideToggle(speed,callback) y animate().

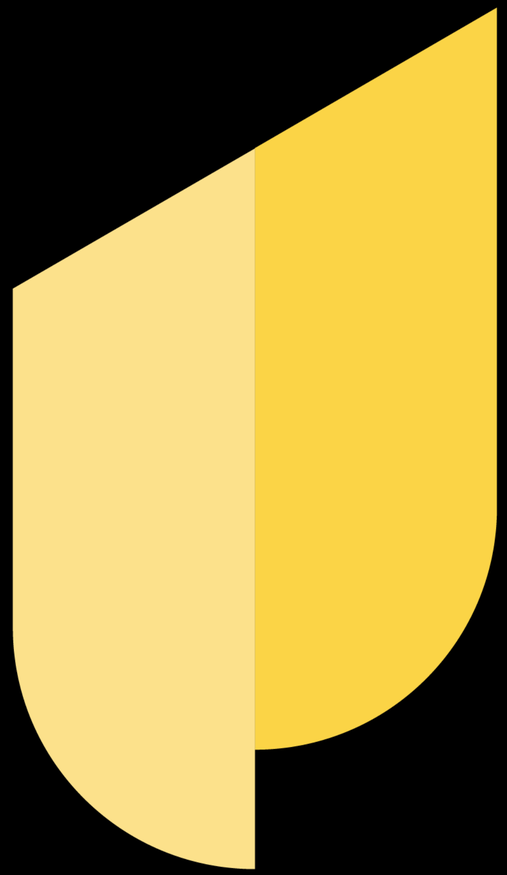
Chaining

- El Chaining es una técnica que consiste en ejecutar múltiples comandos de Jquery, uno tras otro sobre el mismo elemento.

- El siguiente ejemplo aplica color rojo a un párrafo, mediante una animación oculta ese párrafo para finalmente volverlo a mostrar mediante una animación:

  - $("#p1").css("color", "red").slideUp(2000).slideDown(2000);

jQuery



Peticiones Asíncronas

AJAX

- AJAX es el acrónimo de Asynchronous Javascript and XML, es decir, Javascript y XML Asíncrono.
- Este término, se presentó por primera vez en el artículo “Ajax: A New Approach to Web Applications” publicado por Jesse James Garret el 18 de febrero de 2005.
- En resumen, AJAX es una técnica que permite la comunicación asíncrona entre un servidor y un navegador en formato XML mediante programas escritos en Javascript.

Compuesto por:

- **Javascript**: Lenguaje de programación interpretado por los navegadores modernos.
- **XML**: Lenguaje de marcas utilizado para almacenar datos en forma legible. Se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas.
- **Asíncrono**: Tipo de comunicación entre procesos en que quien envía el mensaje continúa con su ejecución sin esperar respuesta del receptor. El tipo de comunicación opuesto es la comunicación síncrona (Quien envía permanece bloqueado esperando a que llegue una respuesta del receptor antes de realizar cualquier otro ejercicio).

Objetivo

- El principal objetivo del AJAX, es intercambiar información entre el servidor y el cliente (navegadores) sin la necesidad de recargar la página. De esta forma, ganamos en usabilidad, experiencia y productividad del usuario final.

Ventajas

- Rapidez en las operaciones.
- Menos carga del servidor (menos transferencia de datos cliente/servidor).
- Menos ancho de banda.
- Soportada por la mayoría de navegadores.
- Interactividad (El usuario no tiene que esperar hasta que lleguen los datos del servidor).
- Portabilidad
- Usabilidad
- Velocidad (Debido a que no hay que recargar la página nuevamente)

Desventajas

- Se pierde el concepto de “volver a la página anterior”.
- Problemas con navegadores antiguos.
- No funciona si el usuario tiene desactivado el Javascript en su navegador.
- Se requieren conocimiento sobre las tecnologías que forman AJAX.
- Problemas SEO, los buscadores no indexan la información recibida vía AJAX.

Método AJAX

- Jquery posee varios métodos para trabajar con Ajax. Sin embargo, todos están basados en el método $.ajax, por lo tanto, su comprensión es obligatoria.

- Generalmente, es preferible utilizar el método $.ajax en lugar de los otros, ya que ofrece más características y su configuración es muy comprensible.

- El método $.ajax es configurado a través de un objeto, el cual contiene todas las instrucciones que necesita jQuery para completar la petición. Dicho método es particularmente útil debido a que ofrece la posibilidad de especificar acciones en caso que la petición haya fallado o no. Además, al estar configurado a través de un objeto, es posible definir sus propiedades de forma separada, haciendo que sea más fácil la reutilización del código

```
$.ajax({
    // la URL para la petición
    url : procesar.php',

    // la información a enviar a la pagina procesar.php
    // (también es posible utilizar una cadena de datos)
    data : { id : 123, nombre : "Ronald Mc Donald" },

    // especifica el tipo de petición si es POST o GET
    type : 'GET',

    // el tipo de información que se espera de respuesta
    dataType : 'json',

    // instrucciones a jecutar si la petición es satisfactoria;
    // la respuesta es pasada como argumento a la función
    success : function(respuesta) {
        $('#mensaje').html(respuesta.mensaje);

    },

    // instrucciones a ejecutar si la petición falla;
    // son pasados como argumentos a la función
    // el objeto de la petición en crudo y código de estatus de la petición
    error : function(xhr, status) {
        alert('A ocurrido un error: ' + status);
    },

    // código a ejecutar sin importar si la petición falló o no
    complete : function(xhr, status) {
        alert('Petición Finalizada);
    }
});
```

Opciones del método $.ajax

- **async** Establece si la petición será asíncrona o no. De forma predeterminada el valor es true. Debe tener en cuenta que si la opción se establece en false, la petición bloqueará la ejecución de otros códigos hasta que dicha petición haya finalizado.

- **cache** Establece si la petición será guardada en la cache del navegador. De forma predeterminada es true para todos los dataType excepto para script y jsonp. Cuando posee el valor false, se agrega una cadena de caracteres anti-cache al final de la URL de la petición.

- **complete** Establece una función de devolución de llamada que se ejecuta cuando la petición esta completa, aunque haya fallado o no. La función recibe como argumentos el objeto de la petición en crudo y el código de estatus de la misma petición.

Opciones del método $.ajax

- **context** Establece el alcance en que la/las funciones de devolución de llamada se ejecutaran (por ejemplo, define el significado de this dentro de las funciones). De manera predeterminada this hace referencia al objeto originalmente pasado al método $.ajax.

- **data** Establece la información que se enviará al servidor. Esta puede ser tanto un objeto como una cadena de datos (por ejemplo foo=bar&baz=bim)

- **dataType** Establece el tipo de información que se espera recibir como respuesta del servidor. Si no se especifica ningún valor, de forma predeterminada, jQuery revisa el tipo de MIME que posee la respuesta.

Opciones del método $.ajax

- **error** Establece una función de devolución de llamada a ejecutar si resulta algún error en la petición. Dicha función recibe como argumentos el objeto de la petición en crudo y el código de estatus de la misma petición.

- **jsonp** Establece el nombre de la función de devolución de llamada a enviar cuando se realiza una petición JSONP. De forma predeterminada el nombre es callback

- **success** Establece una función a ejecutar si la petición a sido satisfactoria. Dicha función recibe como argumentos la información de la petición (convertida a objeto JavaScript en el caso que dataType sea JSON), el estatus de la misma y el objeto de la petición en crudo.

Opciones del método $.ajax

- **error** Establece una función de devolución de llamada a ejecutar si resulta algún error en la petición. Dicha función recibe como argumentos el objeto de la petición en crudo y el código de estatus de la misma petición.

- **jsonp** Establece el nombre de la función de devolución de llamada a enviar cuando se realiza una petición JSONP. De forma predeterminada el nombre es callback

- **success** Establece una función a ejecutar si la petición a sido satisfactoria. Dicha función recibe como argumentos la información de la petición (convertida a objeto JavaScript en el caso que dataType sea JSON), el estatus de la misma y el objeto de la petición en crudo.

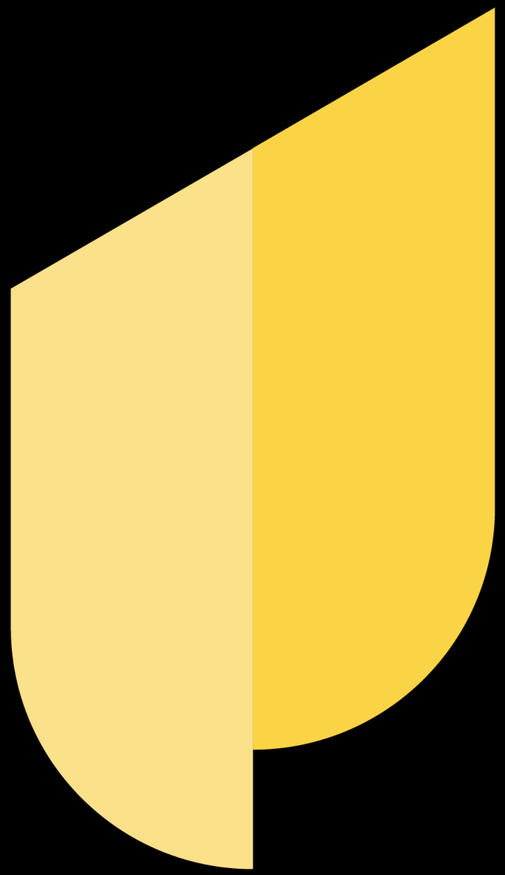
Opciones del método $.ajax

- **timeout** Establece un tiempo en milisegundos para considerar a una petición como fallada. traditional Si su valor es true, se utiliza el estilo de serialización de datos utilizado antes de jQuery 1.4.

- **type** De forma predeterminada su valor es GET. Otros tipos de peticiones también pueden ser utilizadas (como PUT y DELETE), sin embargo pueden no estar soportados por todos los navegadores.

- **url** Establece la URL en donde se realiza la petición. La opción url es obligatoria para el método $.ajax;

Api



Mapas

API

- Una API (siglas de ‘Application Programming Interface’) es un conjunto de reglas (código) y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas: sirviendo de interfaz entre programas diferentes de la misma manera en que la interfaz de usuario facilita la interacción humano-software.

- Las API pueden servir para comunicarse con el sistema operativo (WinAPI), con bases de datos (DBMS) o con protocolos de comunicaciones (Jabber/XMPP). En los últimos años, por supuesto, se han sumado múltiples redes sociales (Twitter, Facebook, Youtube, Flickr, LinkedIn, etc) y otras plataformas online (Google Maps, WordPress…), lo que ha convertido el social media marketing es algo más sencillo, más rastreable y, por tanto, más rentable.

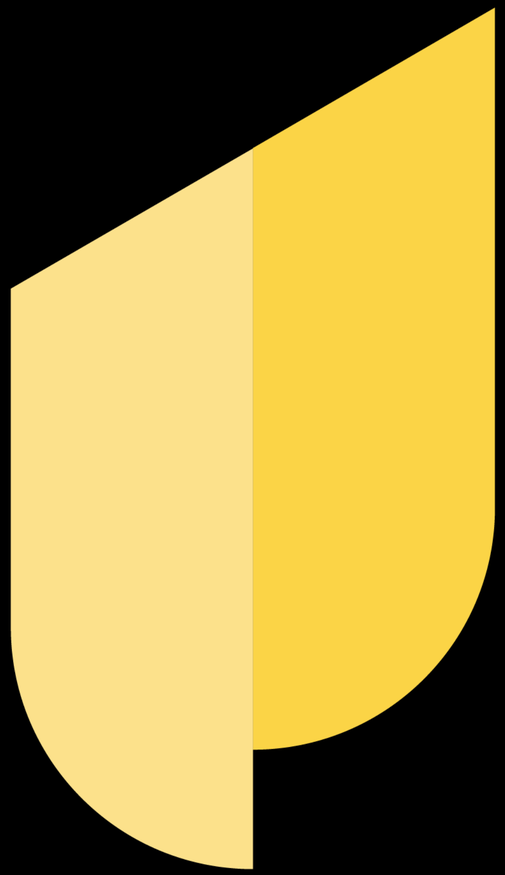
leafletjs.com

- Leaflet es la biblioteca JavaScript de código abierto líder para mapas interactivos compatibles con dispositivos móviles. Con un peso de aproximadamente 39 KB de JS, tiene todas las funciones de mapeo que la mayoría de los desarrolladores necesitan.

- Leaflet está diseñado teniendo en cuenta la simplicidad, el rendimiento y la facilidad de uso. Funciona de manera eficiente en todas las principales plataformas de escritorio y móviles, se puede ampliar con muchos complementos, tiene una API hermosa, fácil de usar y bien documentada y un código fuente simple y legible al que es un placer contribuir.

- https://leafletjs.com/

MUCHAS



GRACIAS