# INSTITUTO SUPERIOR TECNOLÓGICO DEL SUR

## CARRERA

## DISEÑO Y PROGRAMACIÓN WEB

## LENGUAJE WEB 2

## “Eventos JQuery”

**PROFESOR(A) : Amado Cerpa Juan Andrés**

**ALUMNO : Vilca Apaza Christian**

**SEMESTRE : V**

30/03/2021

# Índice

# Eventos con JQuery

## .bind()

Adjunta un controlador a un evento para los elementos.

A partir de jQuery 3.0, .bind()ha quedado obsoleto. Fue reemplazado por el .on()método para adjuntar controladores de eventos a un documento desde jQuery 1.7, por lo que ya se desaconsejaba su uso. Para versiones anteriores, el .bind()método se usa para adjuntar un controlador de eventos directamente a los elementos. Los controladores se adjuntan a los elementos seleccionados actualmente en el objeto jQuery, por lo que esos elementos deben existir en el punto en el que se .bind()produce la llamada . Para una vinculación de eventos más flexible, consulte la discusión sobre la delegación de eventos en .on().

## .blur()

Vincula un controlador de eventos al evento de JavaScript "difuminado" o active ese evento en un elemento.

El blur evento se envía a un elemento cuando pierde el foco. Originalmente, este evento solo se aplicaba a elementos de formulario, como <input>. En navegadores recientes, el dominio del evento se ha ampliado para incluir todos los tipos de elementos. Un elemento puede perder el foco a través de los comandos del teclado, como la tecla Tab, o al hacer clic con el mouse en otra parte de la página.

## .change()

vincula un controlador de eventos al evento de JavaScript "cambiar" o active ese evento en un elemento.
El changeevento se envía a un elemento cuando cambia su valor. Este evento se limita a <input>elementos, <textarea>cajas y <select>elementos. Para las casillas de selección, casillas de verificación y botones de opción, el evento se dispara inmediatamente cuando el usuario realiza una selección con el mouse, pero para los otros tipos de elementos, el evento se aplaza hasta que el elemento pierde el foco.

## .click()

vincula un controlador de eventos al evento de JavaScript "clic" o active ese evento en un elemento.
El click evento se envía a un elemento cuando el puntero del mouse está sobre el elemento y se presiona y suelta el botón del mouse. Cualquier elemento HTML puede recibir este evento

## .contextmenu()

vincula un controlador de eventos al evento de JavaScript "contextmenu" o active ese evento en un elemento.
El contextmenuevento se envía a un elemento cuando se hace clic en él con el botón derecho del mouse, pero antes de que se muestre el menú contextual. En caso de que se presione la tecla del menú contextual, el evento se dispara en el htmlelemento o en el elemento actualmente enfocado. Cualquier elemento HTML puede recibir este evento.

## .dblclick()

vincula un controlador de eventos al evento de JavaScript "dblclick" o active ese evento en un elemento.

El dblclick evento se envía a un elemento cuando se hace doble clic en él. Cualquier elemento HTML puede recibir este evento.

## .delegate()

adjunta un controlador a uno o más eventos para todos los elementos que coincidan con el selector, ahora o en el futuro, en función de un conjunto específico de elementos raíz.

A partir de jQuery 3.0, .delegate()ha quedado obsoleto. Fue reemplazado por el .on()método desde jQuery 1.7, por lo que su uso ya estaba desaconsejado. Sin embargo, para versiones anteriores, sigue siendo el medio más eficaz para utilizar la delegación de eventos.

# .die()

elimine los controladores de eventos adjuntos previamente usando .live()de los elementos.
Cualquier manipulador que se haya adjuntado con .live()puede eliminarse con .die(). Este método es análogo a llamar .off()sin argumentos, que se usa para eliminar todos los controladores adjuntos .on().

A partir de jQuery 1.7 , no se recomienda el uso de .die()(y su método complementario .live()). En su lugar, utilice .off()para eliminar los controladores de eventos vinculados con.on()

## .error()

vincula un controlador de eventos al evento de JavaScript "error".

El evento error se envía a elementos, como imágenes, a los que hace referencia un documento y que el navegador carga. Se llama si el elemento no se cargó correctamente.

## .focus()

Vincula un controlador de eventos al evento de JavaScript "focus" o active ese evento en un elemento.
El evento focus se envía a un elemento cuando gana el foco. Este evento es implícitamente aplicable a un conjunto limitado de elementos, tales como elementos de formulario ( <input>, <select>, etc.) y enlaces ( <a href>). En las versiones recientes del navegador, el evento se puede ampliar para incluir todos los tipos de elementos estableciendo explícitamente la tabindexpropiedad del elemento. Un elemento puede enfocarse a través de comandos de teclado, como la tecla Tab, o haciendo clic en el elemento con el mouse.

## .focusin()

vincula un controlador de eventos al evento "focusin".

El evento focusin se envía a un elemento cuando éste, o cualquier elemento dentro de él, gana el foco. Esto se diferencia del evento de enfoque en que admite la detección del evento de enfoque en los elementos principales (en otras palabras, admite la propagación de eventos).

## .focusout()

vincula un controlador de eventos al evento de JavaScript "focusout".
El evento focusout se envía a un elemento cuando éste, o cualquier elemento dentro de él, pierde el foco. Esto se diferencia del evento de desenfoque en que admite la detección de la pérdida de foco en los elementos descendientes (en otras palabras, admite la propagación de eventos).

## .hover()

Vincula uno o dos controladores a los elementos coincidentes, para que se ejecuten cuando el puntero del mouse ingrese y abandone los elementos.

## .ketdown()

vincula un controlador de eventos al evento de JavaScript "keydown" o active ese evento en un elemento.
El evento keydown se envía a un elemento cuando el usuario presiona una tecla en el teclado. Si se mantiene presionada la tecla, el evento se envía cada vez que el sistema operativo repite la tecla. Se puede adjuntar a cualquier elemento, pero el evento solo se envía al elemento que tiene el foco. Los elementos enfocables pueden variar entre los navegadores, pero los elementos del formulario siempre pueden enfocarse, por lo que son candidatos razonables para este tipo de evento.

## .keypress()

vincula un controlador de eventos al evento de JavaScript de "pulsación de tecla" o active ese evento en un elemento.
El evento keypress se envía a un elemento cuando el navegador registra la entrada del teclado. Esto es similar al keydowncaso, a excepción de que las teclas de modificación y de no impresión, tales como Shift, Escy deletede activación keydownde eventos, pero no keypresseventos. Pueden surgir otras diferencias entre los dos eventos según la plataforma y el navegador.

## .keyup()

vincula un controlador de eventos al evento de JavaScript "keyup" o active ese evento en un elemento.
El evento keyup se envía a un elemento cuando el usuario suelta una tecla del teclado. Se puede adjuntar a cualquier elemento, pero el evento solo se envía al elemento que tiene el foco. Los elementos enfocables pueden variar entre los navegadores, pero los elementos del formulario siempre pueden enfocarse, por lo que son candidatos razonables para este tipo de evento.

## .live()

adjunta un controlador de eventos para todos los elementos que coincidan con el selector actual, ahora y en el futuro.
Este método proporciona un medio para adjuntar controladores de eventos delegados al documentelemento de una página, lo que simplifica el uso de controladores de eventos cuando el contenido se agrega dinámicamente a una página.

## .load()

vincula un controlador de eventos al evento de JavaScript "cargar".
El evento load se envía a un elemento cuando éste y todos los subelementos se han cargado por completo. Este evento se puede enviar a cualquier elemento asociado con una URL: imágenes, scripts, marcos, iframes y el window objeto.

## .mousedown()

vincula un controlador de eventos al evento de JavaScript "mousedown" o active ese evento en un elemento.
El evento mousedown se envía a un elemento cuando el puntero del mouse está sobre el elemento y se presiona el botón del mouse. Cualquier elemento HTML puede recibir este evento.

## .mouseenter ()

vincula un controlador de eventos para que se active cuando el mouse ingresa a un elemento, o activa ese controlador en un elemento.
El evento mouseenter de JavaScript es propiedad de Internet Explorer. Debido a la utilidad general del evento, jQuery simula este evento para que pueda usarse independientemente del navegador. Este evento se envía a un elemento cuando el puntero del mouse ingresa al elemento. Cualquier elemento HTML puede recibir este evento.

## .mouseleave()

vincula un controlador de eventos para que se active cuando el mouse abandona un elemento o activa ese controlador en un elemento.
El evento mouseleave de JavaScript es propiedad de Internet Explorer. Debido a la utilidad general del evento, jQuery simula este evento para que pueda usarse independientemente del navegador. Este evento se envía a un elemento cuando el puntero del mouse abandona el elemento. Cualquier elemento HTML puede recibir este evento.

## .mousemove()

vincula un controlador de eventos al evento de JavaScript "mousemove" o active ese evento en un elemento.
El evento mousemove se envía a un elemento cuando el puntero del mouse se mueve dentro del elemento. Cualquier elemento HTML puede recibir este evento.

## .mouseout()

vincula un controlador de eventos al evento de JavaScript "mouseout" o active ese evento en un elemento.
El evento mouseout se envía a un elemento cuando el puntero del mouse abandona el elemento. Cualquier elemento HTML puede recibir este evento.

## .mouseover ()

vincula un controlador de eventos al evento de JavaScript "mouseover" o active ese evento en un elemento.
El evento mouseover se envía a un elemento cuando el puntero del mouse ingresa al elemento. Cualquier elemento HTML puede recibir este evento.

## .mouseup ()

vincula un controlador de eventos al evento de JavaScript "mouseup" o active ese evento en un elemento.
El evento mouseup se envía a un elemento cuando el puntero del mouse está sobre el elemento y se suelta el botón del mouse. Cualquier elemento HTML puede recibir este evento.

## .on()

Adjunta una función de controlador de eventos para uno o más eventos a los elementos seleccionados
El método on() adjunta controladores de eventos al conjunto de elementos seleccionado actualmente en el objeto jQuery. A partir de jQuery 1.7, el .on()método proporciona todas las funciones necesarias para adjuntar controladores de eventos. Para obtener ayuda en la conversión de los antiguos métodos de evento jQuery,
ver .bind(), .delegate()y .live(). Para eliminar eventos vinculados con .on(),
consulte .off().

## .one()

adjunta un controlador a un evento para los elementos. El controlador se ejecuta como máximo una vez por elemento por tipo de evento.
El .one()método es idéntico a .on(), excepto que el controlador de un elemento y tipo de evento determinados no está vinculado después de su primera invocación.

## .ready()

especifique una función para ejecutar cuando el DOM esté completamente cargado.
El .ready()método ofrece una forma de ejecutar código JavaScript tan pronto como el Modelo de objetos de documento (DOM) de la página sea seguro de manipular. Este suele ser un buen momento para realizar las tareas necesarias antes de que el usuario vea o interactúe con la página, por ejemplo, para agregar controladores de eventos e inicializar complementos. Cuando se agregan varias funciones mediante llamadas sucesivas a este método, se ejecutan cuando el DOM está listo en el orden en que se agregan. A partir de jQuery 3.0, jQuery garantiza que una excepción que se produce en un controlador no impide que se ejecuten los controladores agregados posteriormente.

```
$(function() {
  // Handler for .ready() called.
});
```

## .resize()

vincula un controlador de eventos al evento de JavaScript "redimensionar" o active ese evento en un elemento.

El evento resize se envía al elemento window cuando cambia el tamaño de la ventana del navegador:

## .scroll()

vincula un controlador de eventos al evento de JavaScript "scroll" o active ese evento en un elemento.
El evento scroll se envía a un elemento cuando el usuario se desplaza a un lugar diferente en el elemento.

## .select()
vincula un controlador de eventos al evento de JavaScript "seleccionar" o active ese evento en un elemento.

El selectevento se envía a un elemento cuando el usuario realiza una selección de texto dentro de él. Este evento se limita a campos <input type="text"> y cuadros <textarea>.

## .submit()
vincula un controlador de eventos al evento de JavaScript "enviar" o active ese evento en un elemento.
El evento submit se envía a un elemento cuando el usuario intenta enviar un formulario. Solo se puede adjuntar a elementos <form>. Las formas pueden ser presentadas ya sea haciendo clic en un explícito <input type="submit">, <input type="image">o <button type="submit">, o pulsando Enter cuando ciertos elementos de formulario tienen el foco.

## .toggle()
vincula dos o más controladores a los elementos coincidentes, para que se ejecuten en clics alternativos.

El método .toggle()vincula un controlador para el evento click, por lo que las reglas descritas para la activación de ese click aplican aquí también.

## .trigger()
ejecuta todos los controladores y comportamientos adjuntos a los elementos coincidentes para el tipo de evento dado.

## .unload()
vincula un controlador de eventos al evento de JavaScript "descargar".
El evento unload se envía al elemento window cuando el usuario navega fuera de la página. Esto podría significar una de muchas cosas. El usuario podría haber hecho clic en un enlace para salir de la página o haber escrito una nueva URL en la barra de direcciones. Los botones de avance y retroceso activarán el evento. Cerrar la ventana del navegador hará que se active el evento. Incluso una recarga de la página creará primero un evento unload.

# Conclusión

JQuery esta plagado de funciones todas con utilidades muy buenas que mejoran y facilitan la manera de programar con JavaScript, aunque ahora ya no se necesite mucho jQuery la cantidad de trabajo de tantas funciones sorprende mucho, se debe haber invertido mucho tiempo para crear, refinar y pulir cada función.

# Bibliografía

https://api.jquery.com/category/events/
https://aprende-web.net/librerias/jquery/jquery_8.php
https://www.w3schools.com/jquery/jquery_ref_events.asp
https://es.khanacademy.org/computing/computer-programming/html-js-jquery/dom-events-with-jquery/a/review-dom-events-in-jquery