

Introducción a JQuery

Patricio Martínez

July 26, 2017

Contents

Introducción

JQuery es una librería que nos facilita muchas tareas comunes para nuestra web. Además nos ahorrará gran cantidad de tiempo y esfuerzo ya que nos permite crear muchos efectos y características con muy poco código.

Qué es JQuery

JQuery es una rápida, liviana y con muchas características librería de JavaScript que se basa en el principio de *escribir poco, y hacer mucho*. Permite hacer muchas tareas en nuestro documento sobre manipulación, manejar eventos, añadir animaciones y efectos de manera muy fácil

Ventajas de JQuery

Aquí algunas ventajas que tiene el adoptar JQuery:

- **Salva mucho tiempo** – Te permite salvar mucho tiempo y esfuerzo al usar los selectores y animaciones de JQuery y así te puedes concentrar en el desarrollo.
- Simplifica tareas comunes de JavaScript – JQuery simplifica considerablemente las tareas más comunes de JavaScript. Podemos crear páginas ricas e interactivas con muy pocas líneas de código.
- Fácil de usar - JQuery es muy fácil de usar. Cualquier persona con unos pocos conocimientos de HTML, CSS y JavaScript puede empezar a desarrollar con JQuery
- Compatible con los navegadores – JQuery es compatible con la mayoría de los navegadores modernos
- Absolutamente gratis – Es completamente libre para descargarlo y usarlo

Empezando con JQuery

Para empezar con JQuery vamos a descargar una copia e incluirla en nuestro documento. Podemos descargar dos versiones, una comprimida y otra sin descomprimir.

Una vez descargada la añadimos a nuestro documento en la cabecera tal como ya hemos visto.

Incluyendo JQuery desde CDN

Alternativamente, podemos incluir JQuery en tu página a través de un enlace a CDN (Content Delivery Network), si no queremos descargarlo. CDN nos ofrece la ventaja de reducir los tiempos de carga, ya que ellos tienen hospedaje en multitud de servidores a través de la tierra. Para enlazar JQuery directamente desde CDN añadimos el siguiente código:

```
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
```

Nuestra primera página con JQuery

En esta página vamos a crear un Hola Mundo y le vamos a cambiar el color de la cabecera:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Nuestra primer página con JQuery</title>
  <link rel="stylesheet" type="text/css" href="css/style.css">
  <script src="js/jquery-3.2.1.min.js"></script>
  <script type="text/javascript">
    $(document).ready(function(){
      $("h1").css("color", "#0088ff");
    });
  </script>
</head>
<body>
  <h1>Hola, Mundo!</h1>
</body>
</html>
```

Sintaxis de JQuery

La sintaxis típica en JQuery empieza con el signo de dolar y termina con el punto y coma.

Veamos el siguiente código:

```

<script type="text/javascript">
$(document).ready(function(){
// algun código que ejecutar...
alert("Hola Mundo!");
});
</script>
</head>
<body>
<!--! El contenido va aquí -->
</body>
</html>

```

Explicación del código

Veamos las siguientes partes del código una por una:

- El elemento `<script>` – Como JQuery es una librería, la llamamos desde un archivo externo
- El `$(document).ready(manejador);` – Esto es típicamente conocido como un evento preparado. Básicamente es una función que le pasa al método `ready()` para ser ejecutado tan pronto como el documento empieza a ser manipulado

El método `ready()` es típicamente usado como una función anónima.

Además dentro de la función nosotros escribimos la declaración JQuery que será la que hará la acción con la siguiente sintaxis: `$(selector).acción()`.

Donde, `$(selección)` básicamente selecciona los elementos en el árbol de nodos DOM que puede ser manipulado y `acción()` aplica algunas acciones a los elementos seleccionados como cambiarle las propiedades CSS, cambiar el contenido, etc.

Veamos el siguiente ejemplo:

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Documento JQuery</title>
  <link rel="stylesheet" type="text/css" href="css/style.css">
  <script src="js/jquery-1.11.3.min.js"></script>
  <script type="text/javascript">
    $(document).ready(function(){
      $("p").text("Hola Mundo!");
    });
  </script>
</head>
</html>

```

```

    </script>
</head>
<body>
    <p>No cargado todavía.</p>
</body>
</html>

```

Aquí lo que vemos que hemos seleccionado el elemento `<p>` y con la acción **text** añadimos ese texto. Como vemos text reemplaza el texto. Veamos ahora el siguiente ejemplo:

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Demo con JQuery</title>
    <link rel="stylesheet" type="text/css" href="/examples/css/style.css">
    <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
    <script type="text/javascript">
        $(document).ready(function(){
            $("button").click(function(){
                $("p").text("Hola Mundo!");
            });
        });
    </script>
</head>
<body>
    <p>No saludo al mundo.</p>
    <button type="button">Botón que reemplaza</button>
</body>
</html>

```

Ahora vemos como buscamos el elemento button y la acción se ejecuta con el pulsado del botón y lo que hace es a su vez buscar el elemento `<p>` y sustituir el texto.

Seleccionando elementos con JQuery

JavaScript es comúnmente usado para obtener o modificar contenido o atributos de los elementos HTML o aplicarles algunos efectos, animaciones, etc. Seleccionando elementos a través de JavaScript puede ser doloroso pero con JQuery todo resulta más fácil. La habilidad de hacer de forma simple selecciones de elementos del DOM es una de las características más potentes de JQuery.

Seleccionando elementos por el ID

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando por ID en JQuery</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Cambia el color del elemento con id marca
    $("#marca").css("background", "yellow");
});
</script>
</head>
<body>
    <p id="marca">Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
    <p>Y otro más.</p>
    <p><strong>Nota:</strong> El valor del id debe ser único.</p>
</body>
</html>
```

Seleccionando elementos por la clase

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando por ID en JQuery</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

    $(".marca").css("background", "yellow");
});
</script>
</head>
<body>
```

```

    <p class="marca">Esto es un párrafo.</p>
    <p class="marca">Esto es otro párrafo.</p>
    <p>Y otro más.</p>
</body>
</html>

```

Seleccionando por nombre

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando por ID en JQuery</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Highlight element with id mark
    $("p").css("background", "yellow");
});
</script>
</head>
<body>
    <p>Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
    <div>Y otro más.</div>
</body>
</html>

```

Seleccionando por atributos

Con la palabra reservada **type** podemos elegir el tipo de atributo por el que queremos seleccionar el elemento.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando elementos por atributos</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

```

```

    $('input[type="text"]').css("background", "yellow");
});
</script>
</head>
<body>
    <form>
        <label>Nombre: <input type="text"></label>
        <label>Contraseña: <input type="password"></label>
        <input type="submit" value="Entra">
    </form>
</body>
</html>

```

Seleccionando elementos usando selectores de CSS compuestos

Podemos seleccionar selectores de CSS para hacer nuestra selección más precisa.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando elementos por el selector</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Combinadmos el elemento párrafo con la clase marca
    $("p.mark").css("background", "yellow");

    // Combinamos span con el id marca
    $("#mark span").css("background", "yellow");

    // Combinamos dos elementos ul e li
    $("ul li").css("background", "yellow");

    // Combinamos dos elementos y el id marca
    $("ul#mark li").css("background", "red");

    // Combinamos dos marcas y la clase marca
    $("ul.mark li").css("background", "green");

```

```

    // Highlight all anchor elements with target blank
    $('a[target="_blank"]').css("background", "yellow");
});
</script>
</head>
<body>
    <p>Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
    <p>Otro más.</p>
    <ul>
        <li>Item uno</li>
        <li>Item dos</li>
        <li>Item tres</li>
    </ul>
    <ul id="marca">
        <li>Lista uno</li>
        <li>Lista dos</li>
        <li>Lista tres</li>
    </ul>
    <ul class="marca">
        <li>Y otro</li>
        <li>Pozi</li>
        <li>Pono</li>
    </ul>
    <p>Go to <a href="#">Inicio</a></p>
</body>
</html>

```

Selectores propios de JQuery

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Selectores propios de JQuery</title>
<style type="text/css">
    /* Añadiendo estilo */
    *{
        padding: 5px;
    }

```



```

    }
</style>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Selecciona las filas pares
    $("tr:odd").css("background", "yellow");

    // Selecciona las filas impares
    $("tr:even").css("background", "orange");

    // Selecciona el primer párrafo de un elemento
    $("p:first").css("background", "red");

    // Selecciona el último párrafo de un elemento
    $("p:last").css("background", "green");

    // Selecciona todo lo tipo text dentro de un formulario
    $("form :text").css("background", "purple");

    // Selecciona todo lo tipo password de un formulario
    $("form :password").css("background", "blue");

    // Selecciona todo los input de un formulario
    $("form :submit").css("background", "violet");
});
</script>
</head>
<body>
    <table border="1">
        <thead>
            <tr>
                <th>No.</th>
                <th>Nombre</th>
                <th>Email</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>1</td>

```

```

        <td>Paquito Chocolatero</td>
        <td>paco_elsobrao@mail.com</td>
    </tr>
    <tr>
        <td>2</td>
        <td>Juan Pérez</td>
        <td>juansinmiedo@mail.com</td>
    </tr>
    <tr>
        <td>3</td>
        <td>John Rambo</td>
        <td>johnrambo@mail.com</td>
    </tr>
</tbody>
</table>
<p>Esto es un párrafo.</p>
<p>Esto es otro.</p>
<p>Que cansinos que sois.</p>
<form>
    <label>Nombre: <input type="text"></label>
    <label>Contraseña: <input type="password"></label>
    <input type="submit" value="Sign In">
</form>
</body>
</html>

```

Eventos en JQuery

Los eventos son interacciones del usuario con la página web. JQuery nos ofrece una gran cantidad de métodos para la mayoría de los eventos. Algunos de esos eventos son:

- ready()
- click()
- keypress()
- focus()
- blur()
- change()

- etc

Por ejemplo, el método `ready()` ejecuta algún código cuando el DOM está preparado.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    alert("Hola Mundo!");
});
</script>
</head>
<body>
    El contenido vendrá aquí
</body>
</html>
```

Eventos de ratón

Método `click()`

El método `*click()` de JQuery une una función manejadora de eventos al elemento seleccionado por un evento "click". La función es ejecutado cuando el usuario pulsa el elemento. En el siguiente ejemplo esconderemos un elemento `<p>` de una página cuando son pulsado.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función pulsando un elemento</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<style type="text/css">
    p{
        padding: 20px;
        font: 20px sans-serif;
        background: khaki;
    }
</style>
```

```

</style>
<script type="text/javascript">
$(document).ready(function(){
    $("p").click(function(){
        $(this).slideUp();
    });
});
</script>
</head>
<body>
    <p>Píname y desapareceré.</p>
    <p>Píname y desapareceré.</p>
    <p>Píname y desapareceré.</p>
</body>
</html>

```

El método dblclick()

Este método es igual que el anterior solo que necesita que el usuario haga una doble pulsación sobre el elemento. Veámoslo en el ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función pulsando un elemento</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<style type="text/css">
    p{
        padding: 20px;
        font: 20px sans-serif;
        background: khaki;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("p").dblclick(function(){
        $(this).slideUp();
    });
});

```

```

</script>
</head>
<body>
  <p>Píname y desapareceré.</p>
  <p>Píname y desapareceré.</p>
  <p>Píname y desapareceré.</p>
</body>
</html>

```

El método hover()

El método hover() de JQuery une uno o dos funciones manejadoras de eventos a elementos seleccionados que se ejecutan cuando el puntero del ratón entra y deja los elementos. La primera función es ejecutada cuando el usuario pone el puntero en el elemento y la segunda es cuando el puntero deja el elemento.

En este ejemplo se iluminará el elemento <p> cuando pongas el cursor sobre él y dejará de hacerlo cuando quites el puntero de él.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función con el método hover()</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<style type="text/css">
  p{
    padding: 20px;
    font: 20px sans-serif;
    background: #f2f2f2;
  }
  p.highlight{
    background: red;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  $("p").hover(function(){
    $(this).addClass("highlight");
  }, function(){
    $(this).removeClass("highlight");
  });
});

```

```

});
</script>
</head>
<body>
  <p>Tócame y me pongo rojito.</p>
  <p>Tócame y me pongo rojito.</p>
  <p>Tócame y me pongo rojito.</p>

</body>
</html>

```

El método mouseenter()

Cómo sutilmente dice su propio nombre este método ejecuta la función cuando el puntero entra en el elemento. En el siguiente ejemplo el elemento se iluminará cuando el puntero se pose sobre él.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función con el método mouseenter()</title>
<script src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
<style type="text/css">
  p{
    padding: 20px;
    font: 20px sans-serif;
    background: #f2f2f2;
  }
  p.highlight{
    background: red;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  $("p").mouseenter(function(){
    $(this).addClass("highlight");
  });
  $("p").mouseleave(function(){
    $(this).removeClass("highlight");
  });
});

```

```

    });
});
</script>
</head>
<body>

    <p>Tócame papi.</p>
    <p>Tócame papi.</p>
    <p>Tócame papi.</p>
</body>
</html>

```

El método mouseleave()

Adivínalo tú solo que seguro que los sabes guapi.

Eventos del teclado

El método keypress()

Este método activa la une la función manejadora al elemento seleccionado (normalmente formularios) cuando el navegador recibe entrada del teclado por parte del usuario. El siguiente ejemplo se muestra un mensaje cuando el teclado es pulsado y además cuenta cuantas veces es pulsado el teclado:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función con el evento Keypress</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    p{
        padding: 10px;
        background: lightgreen;
        display: none;
    }
    div{
        margin: 20px 0;
    }
</style>
<script type="text/javascript">

```

```

$(document).ready(function(){
    var i = 0;
    $('input[type="text"]').keypress(function(){
        $("span").text(i += 1);
        $("p").show().fadeOut();
    });
});
</script>
</head>
<body>
    <input type="text">
    <div>Teclas Pulsadas: <span>0</span></div>
    <div><strong>Aviso:</strong> Escribe algo dentro de la caja, anda que te va a gustar....</div>
    <p>DAISYYYYY DAISYYYYY.</p>
</body>
</html>

```

El método keydown()

Es muy parecida al anterior pero la anterior ejecuta la función cuando la tecla es presionada y en esta es cuando la tecla es hundida

El método keyup()

Pues más de lo mismo sólo que ahora la ejecución de la función es cuando la tecla es "soltada"

Eventos de formulario

Los eventos de formularios se activan cuando cuando un control del formulario recibe o pierde el foco o cuando el usuario modifica un valor del formulario como cuando escribe en una caja de entrada, selecciona una selección de un caja de selección, etc. Vamos a ver algunos comunes.

El método change()

El método change() une una función manejadora a un elemento <input>, <textarea> y <select> que es ejecutado cuando el valor es cambiado. En el siguiente ejemplo se mostrará un mensaje de alerta cuando una opción en la caja de selección

```

<!DOCTYPE html>
<html>
<head>

```



```

<meta charset="utf-8">
<title>Ejecutando una función cuando cambia un evento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("select").change(function(){
        var selectedOption = $(this).find(":selected").val();
        alert("Te vas de vacaciones a - " + selectedOption);
    });
});
</script>
</head>
<body>
<h2>¿A qué ciudad vas a ir estas vacaciones?</h2>
    <form>
        <label>Ciudad:</label>
        <select>
            <option>Roma</option>
            <option>Paris</option>
            <option>New York</option>
        </select>
    </form>
    <p><strong>Aviso:</strong> Selecciona un valor del menú desplegable.</p>
</body>
</html>

```

El método focus()

Este método activa una función cuando seleccionamos elementos y gana el foco. En el siguiente ejemplo veremos un mensaje cuando la caja de entrada tenga foco.

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Ejecutando una función cuando un evento tiene foco</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    label{
        display: block;

```

```

        margin: 5px 0;
    }
    label span{
        display: none;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("input").focus(function(){
        $(this).next("span").show().fadeOut("slow");
    });
});
</script>
</head>
<body>
    <form>
        <label>Email: <input type="text"> <span>Fíjate que no te vea nadie</span></label>
        <label>Contraseña: <input type="password"> <span>Cuida tus espaldas</span></label>
        <label><input type="submit" value="Entrar"> <span>Accede ahora</span></label>
    </form>
    <p><strong>Aviso:</strong> Con el ratón o con la tecla "TAB" tendrás foco.</p>
</body>
</html>

```

Método blur()

Ejecuta funciones cuando elementos tales como <input>, <textarea> o <select> pierden el foco.

Método submit()

Este es para formularios, elementos <form> que ejecutan una función cuando el usuario envía un formulario. En el siguiente ejemplo se mostrará un mensaje cuando se vaya a enviar el mensaje.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecuta una función cuando en un formulario se envía éste</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    .error{

```

```

        color: red;
    }
    .success{
        color: green;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("form").submit(function(event){
        var mob = /^[1-9]{1}[0-9]{9}$/;
        var currentValue = $("#inputMobile").val();
        if(mob.test(currentValue) == false && currentValue != 10){
            $("p").html("Número de teléfono inválido number").addClass("error").show().fadeOut(1000);
        } else{
            $("p").html("Número de teléfono válido number").addClass("success").show().fadeOut(1000);
        }
        event.preventDefault();
    });
});
</script>
</head>
<body>
    <form>
        <input type="text" id="inputMobile" maxlength="10" placeholder="Introduce un número de teléfono">
        <input type="submit" value="Submit">
    </form>
</body>
</html>

```

Eventos en el documento/ventana

Aquí los eventos son lanzados cuando un DOM está preparado o cuando la ventana se redimensiona o hay un scroll. Vamos a ver los más usados

El método ready()

Éste ejecuta una función cuando el DOM está completamente cargado. En el siguiente ejemplo se reemplazará un texto cuando el DOM esté cargado.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función con el método ready()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("p").text("El DOM está cargado y preparado para ser manipulado.");
});
</script>
</head>
<body>
    <p>Espera un poquito.</p>
</body>
</html>

```

El método resize()

Este método activa una función cuando la ventana del navegador cambia de tamaño.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función cuando redimensionamos la ventana</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    p{
        padding: 20px;
        font: 20px sans-serif;
        background: #f0e68c;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $(window).resize(function() {
        $(window).bind("redimensiona", function(){
            $("p").text("Altura de la ventana: " + $(window).width() + ", " + " + "Altura de la ventana
        });
    });

```

```

});
});
</script>
</head>
<body>
  <p>Cambia el tamaño de la ventana</p>
</body>
</html>

```

Método scroll()

Esto activa la función cuando detecta que el scroll de un iframe o de la ventana cambia. En este ejemplo se mostrará un mensaje cuando el scroll del navegador cambia.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejecutando una función cuando hacemos scroll</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  p{
    width: 100%;
    padding: 50px 0;
    text-align: center;
    font: bold 34px sans-serif;
    background: #f0e68c;
    position: fixed;
    top: 50px;
    display: none;
  }
  .dummy-content{
    height: 600px;
    font: 34px sans-serif;
    text-align: center;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  $(window).scroll(function() {

```

```

        $("p").show().fadeOut("slow");
    });
});
</script>
</head>
<body>
    <p>Vamos que nos vamos!</p>
    <div class="dummy-content">Volando voy...</div>
    <div class="dummy-content">Volando vengo...</div>
    <div class="dummy-content">Por el camino...</div>
    <div class="dummy-content">Yo me entretengooo.</div>
    <div class="dummy-content">salalaaaaa.</div>
</body>
</html>

```

Efectos en JQuery

Efecto de mostrar y ocultar

Usando los métodos de JQuery **show()** y **hide()** podemos ocultar y mostrar elementos HTML.

El método **hide()** lo que hace simplemente es añadir en el estilo la propiedad **display:none** para ocultar el elemento. Por el contrario, el método **show()** lo que hace es restaurar la propiedad **display** a como estaba.

En este ejemplo podemos ver como funciona:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo del efecto de ocultar de JQuery</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    p{
        padding: 15px;
        background: #F0E68C;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    // Escondemos el párrafo
    $(".hide-btn").click(function(){

```

```

        $("p").hide();
    });

    // Mostramos el párrafo
    $(".show-btn").click(function(){
        $("p").show();
    });
});
</script>
</head>
<body>
    <button type="button" class="hide-btn">Esconde la tontá</button>
    <button type="button" class="show-btn">Muestra la tontá</button>
    <p>Suavementeeee.</p>
    <p>Ocultameeeee.</p>
</body>
</html>

```

También podemos especificar la velocidad a la que queremos que se produzca este efecto. Para ello usaremos o bien las palabras predefinidas **'slow'** y **'fast'** o también podemos indicarlo con exactitud indicando los milisegundos. En el siguiente ejemplo vemos esto:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de un efecto de mostrar/ocultar animado a distintas velocidades</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    p{
        padding: 15px;
        background: #F0E68C;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    // Escondemos los párrafos con distintas velocidades
    $(".hide-btn").click(function(){
        $("p.normal").hide();
        $("p.fast").hide("fast");
    });
});

```

```

        $("p.slow").hide("slow");
        $("p.very-fast").hide(50);
        $("p.very-slow").hide(2000);
    });
    $(".show-btn").click(function(){
        $("p.normal").show();
        $("p.fast").show("fast");
        $("p.slow").show("slow");
        $("p.very-fast").show(50);
        $("p.very-slow").show(2000);
    });
});
</script>
</head>
<body>
    <button type="button" class="hide-btn">Esconde los párrafos</button>
    <button type="button" class="show-btn">Muestra los párrafos</button>
    <p class="very-fast">Este párrafo se mostrará/ocultará muy rápido.</p>
    <p class="normal">Este párrafo se mostrará/ocultará con velocidad por defecto.</p>
    <p class="fast">Este párrafo se mostrará/ocultará de forma rápida.</p>
    <p class="slow">Este párrafo se mostrará/ocultará de forma lenta.</p>
    <p class="very-slow">Este párrafo se mostrará/ocultará de forma muy lenta.</p>
</body>
</html>

```

Al mismo tiempo que ocultamos/mostramos un elemento podemos hacer una llamada a otra función. Veamos este ejemplo que muestra una alerta cuando ocultamos o mostramos un párrafo.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de llamada a otra función mientras mostramos/ocultamos</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    p{
        padding: 15px;
        background: #F0E68C;
    }
</style>

```



```

<script type="text/javascript">
$(document).ready(function(){
    // Muestra un mensaje de alerta después de esconder un párrafo
    $(".hide-btn").click(function(){
        $("p").hide("slow", function(){

            alert("El efecto de ocultación ha terminado.");

        });
    });

    // Muestra un mensaje de alerta después de mostrar un párrafo
    $(".show-btn").click(function(){
        $("p").show("slow", function(){
            alert("El efecto de mostrado ha terminado.");
        });
    });
});
</script>
</head>
<body>
    <button type="button" class="hide-btn">Esconde el párrafo</button>
    <button type="button" class="show-btn">Muestra el párrafo</button>
    <p>Esto es un párrafo.</p>
</body>
</html>

```

Efecto toggle()

Funciona exactamente igual que los métodos `show()` y `hide()`, solo que cambia el estado de oculto a mostrado y de mostrado a oculto.

Efecto fade

Métodos `fadeIn()` y `fadeOut()`

En JQuery tenemos los métodos **`fadeIn()`** y **`fadeOut()`** para mostrar o esconder elementos pero de una incrementando o decrementando su opacidad.

Ejemplo:

```

<!DOCTYPE html>
<html>

```

```

<head>
<meta charset="UTF-8">
<title>Ejemplos del efecto Fade-In y Fade-Out</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  p{
    padding: 15px;
    background: #DDA0DD;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  // Mostrando el párrafo
  $(".out-btn").click(function(){
    $("p").fadeOut();
  });

  // Ocultando el párrafo
  $(".in-btn").click(function(){
    $("p").fadeIn();
  });
});
</script>
</head>
<body>
  <button type="button" class="out-btn">largando párrafo</button>
  <button type="button" class="in-btn">trayendo párrafo</button>
  <p>Esto amigos es una cosa rosa.</p>
  <p>Yo soy otra cosa rosa.</p>
</body>
</html>

```

Al igual que con el método anterior también podemos **controlar la velocidad** de igual manera y también podemos **hacer llamadas** a otras funciones

Método fadeToggle()

Exactamente igual que el anterior pero solo que si el elemento está presente lo oculta y si está oculto lo muestra.

Método fadeTo()

Este método es similar a fadeIn(), pero a diferencia de él fadeTo() nos permite controlar el nivel de opacidad.

```
$(selector).fadeTo(<velocidad>, <opacidad>, <llamada a otra función>);
```

El parámetro de la opacidad es necesario y va entre 0 y 1 y también es requerida la velocidad de la animación.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo del efecto fade con FadeTo()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  p{
    display: none;
    padding: 15px;
    background: #DDA0DD;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  // Fade con diferentes opacidades
  $(".to-btn").click(function(){
    $(".p.none").fadeTo("fast", 0);
    $(".p.partial").fadeTo("slow", 0.5);
    $(".p.complete").fadeTo(2000, 1);
  });
});
</script>
</head>
<body>
  <button type="button" class="to-btn">Fade para esconder los distintos párrafos</button>
  <p class="none">Esto es un párrafo.</p>
  <p class="partial">Esto es otro párrafo.</p>
  <p class="complete">Y párrafo.</p>
</body>
</html>
```

Efecto Sliding (corredero)

Método slideUp() y slideDown()

Los métodos slideUp() y slideDown() sirve para mostrar o esconder un elemento por incremento o decremento de su altura.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de los métodos slideUp() y slideDown()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  p{
    padding: 15px;
    background: #BOC4DE;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  // Efecto de hacia arriba
  $(".up-btn").click(function(){
    $("p").slideUp();
  });

  // Efecto de hacia abajo
  $(".down-btn").click(function(){
    $("p").slideDown();
  });
});
</script>
</head>
<body>
  <button type="button" class="up-btn">Tira parriba</button>
  <button type="button" class="down-btn">Tira pabajo</button>
  <p>Esto es un párrafo.</p>
  <p>Esto es otro párrafo.</p>
</body>
</html>
```

Al igual que los otros métodos vistos hasta ahora podemos controlar la velocidad y hacer llamadas a otras funciones en el proceso.

Método `slideToggle()`

Muestra o oculta un elemento seleccionado con una animación que cambia su altura. En este método también podemos controlar la velocidad y hacer llamadas a otras funciones.

Efectos de animación

Método `animate()`

En JQuery el método `animate()` es usado para crear animaciones. Es usado con propiedades numéricas de CSS como son la altura, anchura, margen, relleno, opacidad, etc. Propiedades no numéricas como pueden ser el color no pueden ser animadas. Para animar color usamos un plugin especial llamado `jquery-color`.

Sintaxis

La sintaxis básica de `animate()` es:

```
$(selector).animate({propiedades}, duración, llamada);
```

Los parámetros de `animate()` tiene el siguiente significado:

- El parámetro propiedad es requerido y define la propiedad CSS que queremos animar.
- La duración es opcional y la podemos especificar con las palabras **'slow'** y **'fast'** o poniendo un número que son los milisegundos.
- La llamada a otra función es opcional

Veamos un ejemplo donde una imagen es animada y va desde una posición inicial hacia la derecha 300 píxeles.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de efectos de animación</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  img{
    position: relative; /* esto es requerido */
```

```

    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("img").animate({
            left: 300
        });
    });
});
</script>
</head>
<body>
    <button type="button">Empieza la animación</button>
    <p>
        
    </p>
</body>
</html>

```

Animando múltiples propiedades a la vez

Con el método `animate()` podemos animar varias propiedades a la vez.

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de una animación de múltiples propiedades</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    .box{
        width: 100px;
        height: 100px;
        background: #9d7ede;
        margin-top: 30px;
        border-style: solid; /* Este valor es requerido */
        border-color: #6f40ce;
    }

```

```

</style>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $(".box").animate({
            width: "300px",
            height: "300px",
            marginLeft: "150px",
            borderWidth: "10px",
            opacity: 0.5
        });
    });
});
</script>
</head>
<body>
    <button type="button">Empieza la animación</button>
    <div class="box"></div>
</body>
</html>

```

Animando múltiples propiedades una a una o en fila.

Podemos animar múltiples propiedades una a una de forma individual usando una característica de encaenamiento de JQuery. Lo veremos en el siguiente capítulo

Animando propiedades con valores relativos

En ocasiones podemos usar valores relativos para animar propiedades. Si un valor es especificado con el prefijo `+=` o `-=` entonces se calcula el valor añadiendo o restando el valor dado al valor actual de la propiedad.

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de una animación con valores relativos</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    .box{

```

```

        width: 100px;
        height: 100px;
        background: #9d7ede;
        margin-top: 30px;
        position: relative; /* Required to move element */
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $(".box").animate({
            top: "+=50px",
            left: "+=50px",
            width: "+=50px",
            height: "+=50px"
        });
    });
});
</script>
</head>
<body>
    <button type="button">Empezar animación</button>
    <div class="box"></div>
</body>
</html>

```

Animando propiedades con valores Pre-definidos

Además de los valores numéricos, cada propiedad puede tomar valores con las palabras 'show', 'hide' y 'toggle'

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de una animación con los valores pre-definidos</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    .box{

```



```

        width: 80%;
        height: 200px;
        background: #9d7ede;
        margin-top: 30px;
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $(".box").animate({
            width: 'toggle'
        });
    });
});
</script>
</head>
<body>
    <button type="button">Animación de cierre</button>
    <div class="box"></div>
</body>
</html>

```

Parada de las animaciones en JQuery

El método stop()

El método stop() es usado para parar las animaciones o efectos que estén funcionando. La sintaxis básica es:

```
$(selector).stop(stopAll, goToEnd);
```

Vamos a explicar los parámetros:

- El booleano *stopAll* es opcional y detiene todas las animaciones. Su valor por defecto es **false** lo que significa que parará la animación que estemos citando en concreto y dejará a las demás.
- El booleano *goToEnd* si especifica si queremos completar la animación de forma inmediata. El valor por defecto es **false**

Veamos un ejemplo que demuestra como funciona el método stop()

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo de parada de una animación con stop()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
    img{
        position: relative; /* propiedad requerida */
    }
</style>
<script type="text/javascript">
$(document).ready(function(){
    // Empieza la animación
    $(".start-btn").click(function(){
        $(".img").animate({left: "+=150px"}, 2000);
    });

    // Stop running animation
    $(".stop-btn").click(function(){
        $(".img").stop();
    });

    // Start animation in the opposite direction
    $(".back-btn").click(function(){
        $(".img").animate({left: "-=150px"}, 2000);
    });

    // Reset to default
    $(".reset-btn").click(function(){
        $(".img").animate({left: "0"}, "fast");
    });
});
</script>
</head>
<body>
    <button type="button" class="start-btn">Empezar</button>
    <button type="button" class="stop-btn">Parar</button>
    <button type="button" class="back-btn">Atrás</button>
    <button type="button" class="reset-btn">Reset</button>

```

```

    <p>
      
    </p>
  </body>
</html>

```

Encadenamiento en JQuery

El método Chaining (encadenamiento)

JQuery ofrece una característica robusta que nos permite unir múltiples acciones para el mismo grupo de elementos, todo en una sola línea de código.

Esto es posible porque los métodos en JQuery devuelven los objetos que pueden ser usados para llamar a otros métodos. Veamos un ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejemplo de un método de encadenamiento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<style type="text/css">
  /* Algunos estilos bonitos */
  p {
    width: 200px;
    padding: 40px 0;
    font: bold 24px sans-serif;
    text-align: center;
    background: #aaccac;
    border: 1px solid #63a063;
    box-sizing: border-box;
  }
</style>
<script type="text/javascript">
$(document).ready(function(){
  $(".start").click(function(){
    $("p").animate({width: "100%"}).animate({fontSize: "46px"}).animate({borderWidth: 30});
  });
  $(".reset").click(function(){
    $("p").removeAttr("style");
  });
});

```

```
});
</script>
</head>
<body>
  <p>Hola fondo norteee!</p>
  <button type="button" class="start">Empezar encadenamiento</button>
  <button type="button" class="reset">Reset</button>
</body>
</html>
```

Hemos escrito todo el encadenamiento en una sola línea:

```
$("p").animate({width: "100%"}).animate({fontSize: "46px"}).animate({borderWidth: 30});
```

Pero si esto es muy embarullado podemos hacerlo en varias líneas

```
$("p")
    .animate({width: "100%"})
    .animate({fontSize: "46px"})
    .animate({borderWidth: 30});
```

Manipulando elementos en JQuery

Getters y Setter en JQuery

Obtener o establecer contenido y valores en JQuery

Algunos métodos en JQuery pueden ser usados para asignar o leer algunos valores en una selección. Unos pocos de esos métodos son **text()**, **html()**, **attr()** y **val()**

Cuando esos métodos son llamados sin argumentos, son referidos como *getters*, porque ellos obtienen (get) o leen los valores de los elementos. Cuando esos métodos son llamados con valores como argumentos, se refiere a ellos como *setters* porque establecer o asignan (set) ese valor.

Método text()

El método **text()** es usado para obtener la combinación de texto de los contenidos seleccionados y sus descendientes, o establecer contenido de texto a los elementos seleccionados.

- Obteniendo contenido de texto con el método **text()**

Vamos a ver como obtenemos el contenido de texto de los párrafos y los mostramos en una cartel de alerta:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Ejemplo cómo obtener texto con text()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $(".btn-one").click(function(){
        var str = $("p").text();
        alert(str);
    });
    $(".btn-two").click(function(){
        var str = $("p:first").text();
        alert(str);
    });
    $(".btn-three").click(function(){
        var str = $("p.extra").text();
        alert(str);
    });
});
</script>
</head>
<body>
    <button type="button" class="btn-one">Obteniendo el texto de todos los párrafos</button>
    <button type="button" class="btn-two">Obteniendo el texto del primer párrafo</button>
    <button type="button" class="btn-three">Obteniendo el texto del último párrafo</button>
    <p>Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
    <p class="extra">Y uno más.</p>
</body>
</html>

```

- Estableciendo contenido de texto con el método text()

En el siguiente ejemplo veremos como el método text() establece contenido en un párrafo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">

```

```

<title>Establecer texto con el métodos text()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $(".btn-one").click(function(){
        $("p").text("Esto es un texto de prueba.");
    });
    $(".btn-two").click(function(){
        $("p:first").text("Esto es otro texto de prueba.");
    });
    $(".btn-three").click(function(){
        $("p.empty").text("Esto es otro texto de prueba.");
    });
});
</script>
</head>
<body>
    <button type="button" class="btn-one">Establece texto en todos los párrafos</button>
    <button type="button" class="btn-two">Establece texto en el primer párrafo</button>
    <button type="button" class="btn-three">Establece texto en el párrafo vacío</button>
    <p>Esto es un párrafo.</p>
    <p>Esto es otro párrafo.</p>
    <p class="empty"></p>
</body>
</html>

```

Método html()

El método html() es usado para obtener o establecer contenido HTML de los elementos.

- Obtener contenido HTML con el método html()

En el siguiente ejemplo se verá como obtener el contenido HTML de los párrafos así como el contenido de un elemento <div>

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Obteniendo contenido HTML de un elemento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>

```

```

<script type="text/javascript">
$(document).ready(function(){
    $(".btn-one").click(function(){
        var str = $("p").html();
        alert(str);
    });
    $(".btn-two").click(function(){
        var str = $("#container").html();
        alert(str);
    });
});
</script>
</head>
<body>
    <button type="button" class="btn-one">Obtener el contenido HTML de un párrafo</button>
    <button type="button" class="btn-two">Obtener el contenido HTML del contenedor</button>
    <div id="container">
        <h1>Hola People!</h1>
        <p>Tengo un <b>pie</b> más grande que otro.</p>
    </div>
</body>
</html>

```

- Estableciendo contenido HTML con el método html()

Veremos en el siguiente ejemplo como establecemos contenido en el elemento <body>

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Estableciendo contenido</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $(".button").click(function(){
        $("body").html("<p>Estoy harto de saludar!</p>");
    });
});
</script>
</head>

```

```

<body>
  <button type="button">Escribe un saludo</button>
</body>
</html>

```

Método attr()

Podemos usar el método attr() para obtener el valor de un atributo o establecerlo

- Obtener el valor de un atributo con el método attr()

En el siguiente ejemplo veremos como obtener el atributo <href> del hiper enlace (elemento <a>) así como el atributo <alt> de un elemento imagen ()

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Obteniendo el valor de un atributo de un elemento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $(".btn-one").click(function(){
    var str = $(".a").attr("href");
    alert(str);
  });
  $(".btn-two").click(function(){
    var str = $(".img#sky").attr("alt");
    alert(str);
  });
});
</script>
</head>
<body>
  <button type="button" class="btn-one">Obtenemos la referencia del enlace</button>
  <button type="button" class="btn-two">Obtenemos el valor del atributo ALT</button>
  <p><a href="https://www.fsf.com/">Free Software Foundation</a></p>
  
</body>
</html>

```


- Establecer el valor de un atributo con el método `attr()`

En el siguiente ejemplo veremos como establecer el atributo **checked** en una caja de comprobación.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Estableciendo un atributo con attr()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $('input[type="checkbox"]').attr("checked", "checked");
    });
});
</script>
</head>
<body>
<p><label><input type="checkbox"></label>Acepto los términos y condiciones y dejo que me d
<button type="button">Comprueba</button>
</body>
</html>
```

El metodo `val()`

Se utiliza el método `val()` para obtener o establecer los valores actuales de elementos de formularios HTML tales como `<input>`, `<select>` y `<textarea>`

- Obteniendo los valores de los campos de un formulario con el método `val()`

Veamos el siguiente ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Obteniendo los valores de un formulario</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button.get-name").click(function(){
```

```

        var name = $("#name").val();
        alert(name);
    });
    $("#button.get-comment").click(function(){
        var comment = $("#comment").val();
        alert(comment);
    });
    $("#button.get-city").click(function(){
        var city = $("#city").val();
        alert(city);
    });
});
</script>
</head>
<body>
    <form>
        <table>
            <tr>
                <td>Nombre:</td>
                <td>
                    <input type="text" id="name">
                </td>
            </tr>
            <tr>
                <td>Comentarios:</td>
                <td>
                    <textarea rows="4" cols="30" id="comment"></textarea>
                </td>
            </tr>
            <tr>
                <td>Ciudad:</td>
                <td>
                    <select id="city">
                        <option>London</option>
                        <option>Paris</option>
                        <option>New York</option>
                    </select>
                </td>
            </tr>
        </table>

```

```

    </form>
    <p><strong>AVISO:</strong> Rellena los siguientes campos.</p>
    <button type="button" class="get-name">Obtén nombre</button>
    <button type="button" class="get-comment">Obtén comentario</button>
    <button type="button" class="get-city">Obtén ciudad</button>
</body>
</html>

```

- Estableciendo los valores de los campos de un formulario con el método val()

Veamos el siguiente ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Estableciendo los valores en un formulario</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        var text = $(this).text();
        $('input[type="text"]').val(text);
    });
});
</script>
</head>
<body>
<h2>Cual es la misión de la NASA que más te gusta?</h2>
    <button type="button">Discovery</button>
    <button type="button">Atlantis</button>
    <button type="button">Endeavour</button>
<p><strong><Aviso:></Aviso:>> Pulsa en los botones de arriba.</p>
    <p>
        <input type="text">
    </p>
</body>
</html>

```

Insertar contenido con JQuery

JQuery dispone de varios métodos como `append()`, `prepend()`, `html()`, `text()`, `before()`, `after()`, `wrap()`, etc, que permite insertar nuevo contenido en un elemento existente.

Como ya hemos visto los métodos `html()` y `text()`, discutiremos los restantes.

Método `append()`

Este método se usa para insertar contenido al final de los elementos seleccionados. En el siguiente ejemplo veremos como se agrega algo de HTML a todos los párrafos y además añadimos contenido al pulsar un botón

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Inserting HTML Contents At the End of the Elements in jQuery</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Append all paragraphs on document ready
    $("p").append(' <a href="#">quiere saber más?</a>');

    // Append a div container on button click
    $("button").click(function(){
        $("#container").append("Esto es un texto de prueba.");
    });
});
</script>
</head>
<body>
    <button type="button">Inserta Texto</button>
    <div id="container">
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam
        <p>Quis quam ut magna consequat faucibus. Pellentesque eget nisi a mi suscipit tincidunt. U
    </div>
</body>
</html>
```

Método prepend()

El método **prepend()** es usado para insertar contenido al principio de los elementos seleccionados.

En el siguiente ejemplo añadiremos contenido al final de los párrafos y cuando pulsemos un botón.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insertando contenido HTML al principio de los elementos seleccionados</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Prepend all paragraphs on document ready
    $("p").prepend("<strong>AVISO:</strong> ");

    // Prepend a div container on button click
    $("button").click(function(){
        $("#container").prepend("Esto es un texto de prueba.");
    });
});
</script>
</head>
<body>
    <button type="button">Inserta Texto</button>
    <div id="container">
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam
        <p>Quis quam ut magna consequat faucibus. Pellentesque eget nisi a mi suscipit tincidunt. U
    </div>
</body>
</html>
```

Insertando múltiples elementos con los métodos append() y prepend()

Tanto append() como prepend() soportan múltiples argumentos como entrada.

En el siguiente ejemplo insertaremos <h1>, <p> y dentro de <body>.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
```

```

<title>Insertar múltiples elementos con append() y prepend()</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        var newHeading = "<h1>Nota importante:</h1>";
        var newParagraph = document.createElement("p");
        newParagraph.innerHTML = "<em>Lorem Ipsum is dummy text...</em>";
        var newImage = $('');
        $("body").append(newHeading, newParagraph, newImage);
    });
});
</script>
</head>
<body>
    <button type="button">Inserta Contenido</button>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam at,
</body>
</html>

```

Método before() y after()

Con estos métodos introducimos contenido antes y después de los elementos seleccionados.

Veamos un ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insertando contenido antes y después de un elemento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Add content after a div container on document ready
    $("#container").after("<p>- The End -</p>");

    // Add content before a div container on document ready
    $("#container").before("<p>- Demo Text -</p>");

    // Add content after heading on button click

```

```

$("button.insert-after").click(function(){
    $("h1").after('');
});

// Add content before heading on button click
$("button.insert-before").click(function(){
    $("h1").before('');
});
});
</script>
<style type="text/css">
    h1{
        display: inline-block;
    }
    body{
        text-align: center;
    }
</style>
</head>
<body>
    <h1>Bienvenidos</h1>
    <hr>
    <button type="button" class="insert-before">Inserta Antes</button>
    <button type="button" class="insert-after">Inserta Después</button>
    <hr>
    <div id="container">
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, varius quam
    </div>
</body>
</html>

```

Método wrap()

El método wrap() es usado para envolver una estructura HTML alrededor de los elementos asociados.

En el siguiente ejemplo envolveremos un div con una clase wrapper y un párrafo con un y un

```

<!DOCTYPE html>
<html>

```

```

<head>
<meta charset="UTF-8">
<title>Ejemplo de envolver un elemento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Envolviendo un div con otro div
    $(".container").wrap('<div class="wrapper"></div>');

    // Envolviendo un párrafo al pulsar un botón
    $("button").click(function(){
        $("p").contents().wrap("<em><b></b></em>");
    });
});
</script>
<style type="text/css">
    .wrapper{
        padding: 20px;
        background: #f0e68c;
        margin: 10px 0;
    }
    .container{
        padding: 15px;
        background: #fff;
        font-size: 24px;
    }
</style>
</head>
<body>
    <button type="button">Púlsame para envolver</button>
    <div class="container">
        <p>Esto es un texto de prueba.</p>
    </div>
</body>
</html>

```

Eliminar contenido con JQuery

JQuery tiene métodos muy útiles tales como `empty()`, `remove()`, `unwrap()`, etc para eliminar elementos o contenido HTML existentes del documento

Método empty()

Este método elimina todos los elementos hijo así como otros descendientes el contenido de texto de un elemento seleccionado desde el DOM.

En el siguiente ejemplo se elimina todo el contenido dentro de los elementos con la clase .container al pulsar un botón.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Eliminando contenido</title>
<style type="text/css">
.container{
    padding: 10px;
    background: #f0e68c;
    border: 1px solid #bead18;
}
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

    $("button").click(function(){
        $(".container").empty();
    });
});
</script>
</head>
<body>
    <div class="container">
        <h1>Buenos días por la mañana!</h1>
        <p class="hint"><strong>Aviso:</strong> Si pulsas el botón lo borrarás todo incluido el bot
        <button type="button">Vaciar contenedor</button>
    </div>
</body>
</html>
```

El método remove()

Este método elimina los elementos seleccionados y todo lo que hay dentro de él.

En el siguiente ejemplo borraremos todos los elementos `<p>` con la clase `.hint`. Los elementos dentro de ese elemento también serán borrados

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Eliminando elementos desde el DOM</title>
<style type="text/css">
.container{
    padding: 10px;
    background: #f0e68c;
    border: 1px solid #bead18;
}
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    // Borra el párrafo con la clase hint
    $("button").click(function(){
        $("p.hint").remove();
    });
});
</script>
</head>
<body>
    <div class="container">
        <h1>Saluditos vecinitos!</h1>
        <p class="hint"><strong>Aviso:</strong> Si pulsas este botón borrarás el párrafo.</p>
        <button type="button">Borra el párrafo</button>
    </div>
</body>
</html>
```

El método `unwrap()`

Elimina todo lo que hay alrededor de un elemento. Hace lo contrario que el método `wrap()`

El método `removeAttr()`

Elimina los atributos de un elemento seleccionando.

En el siguiente ejemplo borraremos el atributo href de un elemento <a> al pulsar un botón

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Eliminando un atributo</title>
<style type="text/css">
  a{
    font-size: 18px;
    margin-right: 20px;
  }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  // Eliminamos el atributo href
  $("button").click(function(){
    $("a").removeAttr("href");
  });
});
</script>
</head>
<body>
  <div class="container">
    <p>
      <a href="">Inicio</a>
      <a href="">Sobre</a>
      <a href="">Contacto</a>
    </p>
    <button type="button">Borra atributos</button>
  </div>
</body>
</html>
```

Clases CSS con JQuery

JQuery ofrece métodos para la manipulación de clases tales como `addClass()`, `removeClass()`, `toggleClass()`, etc.

Método addClass

Este método añade una o más clases al elemento seleccionado

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Añadiendo clases a un elemento</title>
<style type="text/css">
    .page-header{
        color: red;
        text-transform: uppercase;
    }
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("h1").addClass("page-header highlight");
    });
});
</script>
</head>
<body>
    <h1>Saluditos</h1>
    <p>Ganicas locas de que no haga tanto calor.</p>
    <button type="button">Añade clases</button>
</body>
</html>
```

Método removeClass()

Elimina clases de los elementos seleccionados

Ejemplo:

```
<!DOCTYPE html>
<html>
```

```

<head>
<meta charset="utf-8">
<title>Eliminando clases en JQuery</title>
<style type="text/css">
    .page-header{
        color: red;
        text-transform: uppercase;
    }
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("h1").removeClass();
        $("p").removeClass();
    });
});
</script>
</head>
<body>
    <h1 class="page-header">Texto de prueba</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    <p class="hint highlight"><strong>Nota:</strong> Lorem Ipsum is dummy text.</p>
    <button type="button">Eliminando clases</button>
</body>
</html>

```

Método toggleClass()

Este método añade o borra según sea la selección clases de los elementos seleccionados

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Cambiano clases con JQuery</title>

```

```

<style type="text/css">
  p{
    padding: 10px;
    cursor: pointer;
    font: bold 16px sans-serif;
  }
  .highlight{
    background: yellow;
  }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("p").click(function(){
    $(this).toggleClass("highlight");
  });
});
</script>
</head>
<body>
  <p>Pinchame y me pondré rojo o no.</p>
  <p class="highlight">Pinchame y me pondré rojo o no.</p>
  <p>Pinchame y me pondré rojo o no.</p>
</body>
</html>

```

Propiedades de estilo con JQuery

JQuery también nos ofrece los métodos para obtener y establecer propiedades CSS

Método `css()`

Este es un método rápido para acceder a los estilos de los elementos HTML

- Obtener el valor actual CSS

Podemos obtener el valor de la propiedad CSS de un elemento simplemente pasando el nombre de la propiedad como un parámetro al método `css()`. Aquí vemos la sintaxis básica:

```
$(selector).css("nombrePropiedad");
```

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Obteniendo valores de una propiedad CSS</title>
<style type="text/css">
    div{
        width: 100px;
        height: 100px;
        margin: 10px;
        cursor: pointer;
        display: inline-block;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("div").click(function(){
        var color = $(this).css("background-color");
        $("#result").html(color);
    });
});
</script>
</head>
<body>
    <div style="background-color:orange;"></div>
    <div style="background-color:#ee82ee;"></div>
    <div style="background-color:rgb(139,205,50);"></div>
    <div style="background-color:#f00;"></div>
    <p>El valor del color de fondo es: <b id="result"></b></p>
</body>
</html>

```

- Establecer una propiedad CSS y su valores

El método `css()` pueden tomar el nombre de una propiedad y un valor como parámetros separados para establecerlos en una propiedad CSS para el elemento. Veamos el siguiente ejemplo:

```

<!DOCTYPE html>
<html>
<head>

```

```

<meta charset="utf-8">
<title>Estableciendo valores de una propiedad CSS</title>
<style type="text/css">
    .box{
        width: 100px;
        height: 100px;
        margin: 10px;
        cursor: pointer;
        border: 1px solid #cdcdcd;
        display: inline-block;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $(".box").click(function(){
        $(this).css("background-color", "red");
    });
});
</script>
</head>
<body>
    <div class="box"></div>
    <div class="box"></div>
    <div class="box"></div>
    <div class="box"></div>
    <p><strong>Aviso:</strong> Pulsa dentro de la caja vacía para llenarla con el color de fondo
</body>
</html>

```

- Establecer múltiples propiedades CSS

También podemos establecer múltiples valores CSS con el método `css()`. La sintaxis básica es:

```
$(selector).css({"nombrePropiedad": "valor", "nombrePropiedad": "valor", ...})
```

Veamos el siguiente ejemplo:

```

<!DOCTYPE html>
<html>
<head>

```



```


<meta charset="utf-8">
<title>Establecer el valor CSS de múltiples propiedades</title>
<style type="text/css">
    p{
        font-size: 18px;
        font-family: Arial, sans-serif;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("p").css({"background-color": "yellow", "padding": "20px"});
    });
});
</script>
</head>
<body>
    <h1>Saludos desde la tierra</h1>
    <p style="background-color:orange;">Esto es un párrafo.</p>
    <p style="background-color:#ee82ee;">Esto es otro párrafo.</p>
    <p style="background-color:rgb(139,205,50);">Y otro más.</p>
    <p>El último párrafo.</p>
    <button type="button">Añadiendo estilo CSS</button>
</body>
</html>

```

Dimensiones en JQuery

Vamos a aprender como obtener y establecer dimensiones en una caja tal como altura o anchura.

JQuery ofrece varios métodos, tales como `height()`, `innerHeight()`, `outerHeight()`, `width()`, `innerWidth()` y `outerWidth()` para obtener y establecer dimensiones CSS para los elementos. En la siguiente ilustración entenderemos como esos métodos calculan la s dimensiones de una caja.



`./img/dimensiones-jquery.png`

Métodos `width()` y `height()`

Los métodos `width()` y `height()` obtienen o establecer la altura y la anchura de los elementos. La altura y anchura no incluye relleno (`padding`), bode (`border`) y margen (`margin`) de el elemento. El siguiente ejemplo nos devolverá la altura y anchura de un elemento `div`

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Obteniendo la altura y anchura de un elemento</title>
<style type="text/css">
    #box{
        width: 300px;
        height: 200px;
        padding: 25px;
        text-align: justify;
        border: 10px solid #c6b51a;
        background: #f0e68c;
        margin: 15px;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        var divWidth = $("#box").width();
        var divHeight = $("#box").height();
        $("#result").html("Altura: " + divWidth + ", " + "Anchura: " + divHeight);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Obtén la altura y anchura</button>
    <p id="result"></p>
</body>
</html>

```

De manera similar podemos establecer al altura y anchura de los elementos incluyendo los parámetros en los métodos `width()` y `height()`. El valor puede ser una cadena de texto (número y unidad. ej: 100px, 20em) o un número. En el siguiente ejemplo estableceremos la altura de un elemento `<div>` a 400 píxeles y una anchura de 300 píxeles respectivamente.

```

<!DOCTYPE html>
<html>

```

```

<head>
<meta charset="utf-8">
<title>Estableciendo la altura y anchura de un elemento</title>
<style type="text/css">
    #box{
        width: 300px;
        height: 200px;
        padding: 25px;
        text-align: justify;
        border: 10px solid #c6b51a;
        background: #f0e68c;
        margin: 15px;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#button").click(function(){
        $("#box").width(400).height(300);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Establece al altura y la anchura</button>
</body>
</html>

```

Métodos innerWidth() y innerHeight()

Los métodos de JQuery innerWidth() e innerHeight() obtienen o establecen el interior de la altura y la anchura de los elementos respectivamente. El interior incluye el relleno pero excluye el borde y el margen de los elementos. En el siguiente ejemplo obtendremos el valor de altura y anchura de un elemento <div>

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Obteniendo la altura y anchura interiores</title>

```

```

<style type="text/css">
  #box{
    width: 300px;
    height: 200px;
    padding: 25px;
    text-align: justify;
    border: 10px solid #c6b51a;
    background: #f0e68c;
    margin: 15px;
  }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("#button").click(function(){
    var divWidth = $("#box").innerWidth();
    var divHeight = $("#box").innerHeight();
    $("#result").html("Inner Width: " + divWidth + ", " + "Inner Height: " + divHeight);
  });
});
</script>
</head>
<body>
  <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
  <button type="button">Obten la altura y anchura interior</button>
  <p id="result"></p>
  <hr>
  <p><strong>Aviso:</strong><b>innerWidth()</b> incluye las propiedades (<b>width</b> + <b>padding-
</body>
</html>

```

De igual manera, podemos establecer al altura y anchura interiores. Al alterar el `innerWidth()` y el `innerHeight()` también alteramos la altura y la anchura del área de contenido.

Por ejemplo, si la actual altura de un elemento es 300 píxeles y la suma de el relleno izquierdo y derecho es igual a píxeles entonces la nueva altura del elemento para obtener una altura de 4000 será de 350, es decir, **la nueva altura = altura interior - relleno horizontal**. Se puede estimar lo mismo con la altura.

```

<!DOCTYPE html>
<html>
<head>

```

```

<meta charset="utf-8">
<title>Establecer alturas y anchuras interiores</title>
<style type="text/css">
    #box{
        width: 300px;
        height: 200px;
        padding: 25px;
        text-align: justify;
        border: 10px solid #c6b51a;
        background: #f0e68c;
        margin: 15px;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#button").click(function(){
        $("#box").innerWidth(400).innerHeight(300);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Establecer altura y anchura interiores</button>
</body>
</html>

```

Métodos `outerWidth()` y `outerHeight()`

Los métodos `outerWidth()` y `outerHeight()` sirven para establecer u obtener la altura y anchura exterior. Esto incluye el relleno y el borde pero excluye el margen del elemento. En el siguiente ejemplo devolverá la altura y anchura exteriores de un elemento `<div>Y`

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Obteniendo la altura y anchura exteriores</title>
<style type="text/css">

```

```

#box{
    width: 300px;
    height: 200px;
    padding: 25px;
    text-align: justify;
    border: 10px solid #c6b51a;
    background: #f0e68c;
    margin: 15px;
}
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#button").click(function(){
        var divWidth = $("#box").outerWidth();
        var divHeight = $("#box").outerHeight();
        $("#result").html("Outer Width: " + divWidth + ", " + "Outer Height: " + divHeight);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Obtén la altura y anchura exteriores</button>
    <p id="result"></p>
    <hr>
    <p><strong>Aviso:</strong><b>outerWidth()</b> incluye las propiedades CSS (<b>width</b> + <b>padd
</body>
</html>

```

Podemos obtener también la altura y anchura incluyendo relleno y borde al igual que el margen del elemento. Para ello añadiremos el parámetro **true** al método tal como `outerWidth(true)` y `outerHeight(true)`

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Obteniendo la altura y anchura con el margen</title>
<style type="text/css">

```

```

#box{
    width: 300px;
    height: 200px;
    padding: 25px;
    text-align: justify;
    border: 10px solid #c6b51a;
    background: #f0e68c;
    margin: 15px;
}
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#button").click(function(){
        var divWidth = $("#box").outerWidth(true);
        var divHeight = $("#box").outerHeight(true);
        $("#result").html("Outer Width: " + divWidth + ", " + "Outer Height: " + divHeight);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Obten la altura y anchura exteriores con el margen</button>
    <p id="result"></p>
    <hr>
    <p><strong>Note:</strong><b>outerWidth(true)</b> incluye (<b>width</b> + <b>padding-left</b> + <b>
</body>
</html>

```

De igual manera podemos establecer la altura y anchura exteriores añadiendo el valor como parámetro a los métodos. Ahora el calculo sería el siguiente para la anchura:

- **Altura - (Relleno horizontal + borde horizontal)**

Ejemplo:

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">

```



```

<title>Estableciendo la altura y anchura exteriores</title>
<style type="text/css">
    #box{
        width: 300px;
        height: 200px;
        padding: 25px;
        text-align: justify;
        border: 10px solid #c6b51a;
        background: #f0e68c;
        margin: 15px;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("#box").outerWidth(400).outerHeight(300);
    });
});
</script>
</head>
<body>
    <div id="box">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eu sem tempor, variu
    <button type="button">Establece la altura y anchura exteriores</button>
</body>
</html>

```

JQuery avanzado

JQuery Traversing (recorrido)

Qué es el traversing

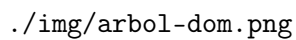
Los selectores de JQuery nos permiten seleccionar los elementos recorriendo el árbol DOM. Pero en muchas ocasiones necesitamos seleccionar un elemento padre y uno antepasado. Ahí es donde entra en juego el método de recorrido por el DOM. Con esos método por demos ir hacia arriba, abajo y alrededor del árbol DOM muy fácilmente.

El recorrido DOM es una muy prominente característica de JQuery. Para hacer esto necesitamos entender las relaciones entre los elementos de un árbol DOM.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Ejemplo de árbol DOM</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
</head>
<body>
  <div class="container">
    <h1>Hola Mundo</h1>
    <p>Esto es <em>un simple párrafo</em>.</p>
    <ul>
      <li>Item Uno</li>
      <li>Item Dos</li>
    </ul>
  </div>
</body>
</html>
```

El código HTML de este ejemplo lo podemos representar como este árbol DOM:



./img/arbol-dom.png

El diagrama anterior muestra las relaciones padre/hijo entre los elementos:

- El elemento `<body>` es **padre** del elemento `<div>` y es el antepasado de todo. El elemento `<div>` es **padre** de `<h1>`, `<p>` y ``, e **hijo** del elemento `<body>`.
- El elemento `<h1>`, `<p>` y `` son **hermanos**, desde que tienen el mismo padre.

- El elemento `<h1>` es un **hijo** del elemento `<div>` y un **descendiente** del elemento `<body>`. Este elemento no tiene ningún hijo
- El elemento `<p>` es el padre del elemento ``, **hijo** del elemento `<div>` y **descendiente** del elemento `<body>`. El elemento `` es un **hijo** del elemento `<p>` y un **descendiente** de los elementos `<div>` y `<body>`

Recorriendo el árbol DOM

Ahora que hemos entendido las relaciones lógicas entre los elementos en un árbol DOM. En los siguientes capítulos aprenderemos como hacer operaciones para recorrer el árbol hacia arriba y hacia abajo usando JQuery

Antepasados

Recorriendo el árbol DOM hacia arriba

En las relaciones lógicas un antepasado es un padre, abuelo, bisabuelo y así.

JQuery dispone de métodos muy útiles tales como **parent()**, **parents()** y **parentsUntil()** que pueden usarse para recorrer el árbol DOM hacia arriba entre uno o múltiples niveles.

Método parent()

El método **parent()** es usado para obtener directamente el padre del elemento seleccionado.

En el siguiente ejemplo resaltaremos el padre directo del elemento `` es cual es `` añadiendo la clase `.highlight`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando el padre de un elemento</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("li").parent().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
```

```

    <p>Esto es <em>un simple párrafo</em>.</p>
    <ul>
        <li>Item Uno</li>
        <li>Item Dos</li>
    </ul>
</div>
</body>
</html>

```

El método parents()

El método **parents()** es usado para obtener los antepasados de un elemento seleccionado. En el siguiente ejemplo añadiremos un borde alrededor de todos los elementos antepasados del elemento ``, los cuales son ``, `<div>`, `<body>` y `<html>`

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando todos los antepasados de un elemento</title>
<style type="text/css">
    *{
        margin: 10px;
    }
    .frame{
        border: 2px solid green;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("li").parents().addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>

```

```

        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>

```

Podemos opcionalmente incluir uno o más selectores como parámetros en el método `parents()` para filtrar la búsqueda de los antepasados. En el siguiente ejemplo aplicaremos un borde al antepasado de `` que son los elementos `<div>`.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un antepasado específico</title>
<style type="text/css">
    *{
        margin: 10px;
    }
    .frame{
        border: 2px solid green;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("li").parents("div").addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es<em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>

```

```
    </div>
</body>
</html>
```

Método parentsUntil()

Este método se usa para obtener todos los antepasados sin incluir el elemento que coincida con el selector. En palabras simples podemos pedir que nos devuelva todos los elementos antepasados entre dos elementos dados.

En el siguiente ejemplo vamos a añadir un bode alrededor de todos los elementos antepasados de excluyendo <html>, es decir, añadiremos un borde a , <div> y <body>

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Selecting All the Ancestors between Two Elements in jQuery</title>
<style type="text/css">
    *{
        margin: 10px;
    }
    .frame{
        border: 2px solid green;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("li").parentsUntil("html").addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
```

```
    </div>
</body>
</html>
```

Descendientes

Recorriendo el árbol DOM hacia abajo

En las relaciones lógicas un descendiente es un hijo, nieto, gran nieto, y así.

JQuery provee de métodos muy útiles tales como **children()** y **find()** que se pueden usar para recorrer hacia abajo por el árbol DOM uno o varios niveles de manera muy fácil y encontrar u obtener hijos u otros descendientes de un elemento en la jerarquía.

Método children()

El método children() es usado para obtener directamente el hijo de un elemento. En el siguiente ejemplo resaltaremos el hijo directo del elemento el cual es añadiendo la clase .highlight.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando hijos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").children().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
```



```

        <li>Item Dos</li>
    </ul>
</div>
</body>
</html>

```

El método find()

El método find() es usado para obtener los descendientes de un elemento seleccionado.

Los métodos find() y children() son similares, excepto que find() busca a través de múltiples niveles en el árbol hasta el último descendiente, mientras que children() solo lo hace un nivel. En el siguiente ejemplo añadiremos un borde alrededor de todos los elementos que sean descendientes del elemento <div>.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando hijos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").find("li").addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>

```

</html>

Sin embargo, si queremos obtener todos los descendientes podemos usar el selector universal

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando hijos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").find("*").addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>
```

Hermanos

Recorriendo el árbol DOM lateralmente

En las relaciones lógicas los hermanos son los elementos que tienen el mismo padre.

JQuery provee de varios métodos tales como **siblings()**, **nest()**, **nestAll()**, **nextUntil()**, **prev()**, **prevAll()** y **prevUntil()** que podemos usar.

Método siblings()

El método es usado para obtener el elemento hermano de un elemento seleccionado. En el siguiente ejemplo resaltaremos el hermano de <p> que son <h1> y .

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando hermanos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("p").siblings().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>
```

Opcionalmente podemos incluir uno o más selectores como parámetros en el método siblings() para filtrar nuestra búsqueda. En el siguiente ejemplo aplicaremos un borde alrededor de los hermanos de <p> que sea elementos .

```
<!DOCTYPE html>
<html>
<head>
```

```

<meta charset="utf-8">
<title>Seleccionando hermanos específicos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("p").siblings("ul").addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>

```

Método next()

El método next() es usado para para obtener el siguiente hermano del elemento seleccionado. En el siguiente ejemplo vamos a resaltar el próximo hermano de <p> que es .

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando el siguiente hermano con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }

```

```

</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("p").next().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>

```

Método nextAll()

Este método es usado para obtener todos los hermanos que siguen del elemento seleccionado. En el siguiente ejemplo resaltaremos todos los hermanos de <p>

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando hijos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("p").nextAll().addClass("highlight");
});

```

```

</script>
</head>
<body>
  <div class="container">
    <h1>Hola Mundo</h1>
    <p>Esto es <em>un simple párrafo</em>.</p>
    <ul>
      <li>Item Uno</li>
      <li>Item Dos</li>
    </ul>
  </div>
</body>
</html>

```

El método nextUntil()

Este método es usado para obtener todos los hermanos que sigan sin incluir el elemento que coincide con el selector.

En el siguiente ejemplo resaltaremos los hermanos de <h1> excluyendo

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando todos los elementos hermanos entre dos elementos</title>
<style type="text/css">
  .highlight{
    background: yellow;
  }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("h1").nextUntil("ul").addClass("highlight");
});
</script>
</head>
<body>
  <div class="container">
    <h1>Hola Mundo</h1>

```

```

    <p>Esto es <em>un simple párrafo</em>.</p>
    <ul>
        <li>Item Uno</li>
        <li>Item Dos</li>
    </ul>
</div>
</body>
</html>

```

El método prev()

Este método es usado para obtener el elemento hermano que inmediatamente precede. En el siguiente ejemplo resaltaremos el hermano previo de el cual es el elemento <p>.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando el hermano previo en JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").prev().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>

```

```
</body>
</html>
```

El método prevAll()

Con el obtenemos todos los hermanos que preceden al elemento seleccionado.

En el siguiente ejemplo resaltaremos a todos los hermanos que preceden a

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando a todos los hermanos con JQuery</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").prevAll().addClass("highlight");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>
```

Método prevUntil()

Este método es usado para seleccionar a todos los hermanos previos hasta el que coincida con el selector.

En el siguiente ejemplo vamos a resaltar todos los elementos hermanos previos de `` excluyendo al elemento `<h1>`.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando a los hermanos previos entre dos elementos</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul").prevUntil("h1").addClass("frame");
});
</script>
</head>
<body>
    <div class="container">
        <h1>Hola Mundo</h1>
        <p>Esto es <em>un simple párrafo</em>.</p>
        <ul>
            <li>Item Uno</li>
            <li>Item Dos</li>
        </ul>
    </div>
</body>
</html>
```

Filtrando elementos

Filtrando elementos en la selección

JQuery proporciona varios métodos tales como `filter()`, `first()`, `last()`, `eq()`, `slice()`, `has()`, `not()`, etc, que permiten restringir la búsqueda de elementos en el árbol DOM.

Método first()

Este método filtra los elementos coincidentes y devuelve el primero de ellos. En el siguiente ejemplo vamos a resaltar el primer elemento dentro del elemento

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando el primer elemento</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").first().addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
</body>
</html>
```

Método last()

Este método devuelve el último de los elementos coincidentes. Ahora vamos a hacer lo mismo que en el ejemplo anterior solo que obteniendo el último elemento que va a ser el que vamos a resaltar.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando el último elemento</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").last().addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
</body>
</html>
```

Método eq()

El método eq() devuelve un único elemento especificado con un número de una serie de elementos que coinciden con la búsqueda.

En el siguiente ejemplo vamos a resaltar el segundo que están dentro de un elemento

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un elemento definido por un índice</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").eq(1).addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
</body>
</html>
```

También se puede especificar un **número negativo** lo que indica que empieza a contar desde el final.

Método filter()

Este método puede tomar el selector o una función como argumentos para filtrar un conjunto de elementos siguiendo un criterio específico.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un elemento siguiendo un criterio con filter()</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").filter(:even).addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
</body>
</html>
```

```
</ul>
</body>
</html>
```

En el siguiente ejemplo vamos a obtener los elementos que hayan dentro de y que sean pares los cuales vamos a resaltar.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un elemento siguiendo un criterio con filter()</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").filter(function(index){
        return index % 2 !== 0;
    }).addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
```

```

        <li>Cuarto elemento</li>
    </ul>
</body>
</html>

```

Método has()

El método has() filtra el conjunto de elementos coincidentes y devuelve solo esos elementos que tienen un específico elemento descendiente. En el siguiente ejemplo vamos a resaltar todos los elementos que tengan elementos como descendientes.

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando elementos con una descendencia específica</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").has("ul").addClass("highlight");
});
</script>
</head>
<body>
    <ul>
        <li>Sección 1</li>
        <li>Sección 2</li>
        <li>
            <ul>
                <li>Sección 2.1</li>
                <li>Sección 2.2</li>
                <li>Sección 2.3</li>
            </ul>
        </li>
        <li>Sección 4</li>
    </ul>

```

```
</ul>
</body>
</html>
```

El método not()

Este método filtra un conjunto de elementos coincidentes y devuelve todos los elementos que están en las codiciones especificadas. Puede ser un selector o una función.

Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un elemento que no siga un criterio</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").not(":even").addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
```



```

        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
</body>
</html>

```

El método `not()` también puede tomar funciones como argumentos al igual que `filter()`

Método `slice()`

El método `slice()` filtra un conjunto de elementos coincidentes y devuelve un rango de índices. El método acepta número de comienzo y final (opcional) como argumentos.

En el siguiente ejemplo resaltaremos el primero y el segundo elemento `` dentro de ``

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Seleccionando un elemento siguiendo un criterio con filter()</title>
<style type="text/css">
    .highlight{
        background: yellow;
    }
</style>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("ul li").slice(0, 2).addClass("highlight");
});
</script>
</head>
<body>
    <h2>Lista desordenada</h2>
    <ul>
        <li>Primer elemento</li>
        <li>Segundo elemento</li>
        <li>Tercer elemento</li>
        <li>Cuarto elemento</li>
    </ul>
    <hr>
    <h2>Otra lista desordenada</h2>

```

```
<ul>
  <li>Primer elemento</li>
  <li>Segundo elemento</li>
  <li>Tercer elemento</li>
  <li>Cuarto elemento</li>
</ul>
</body>
</html>
```

El método `slice()` también puede recibir números negativos, que indica que empezamos a contar posiciones desde el final.

Ajax en JQuery

Qué es Ajax

Ajax son las siglas de Asynchronous Javascript and XML. Ajax solo significa que se cargan datos desde el servidor hacia el navegador web sin necesidad de recargar la página completamente.

Básicamente, con Ajax se hace uso de un objeto basado en JavaScript llamado `XMLHttpRequest` que envía y recibe información a y desde un servidor web de manera asíncrona, en segundo plano, sin interferir en la experiencia de usuario.

Ajax con JQuery

Diferentes navegadores implementan Ajax de manera diferente lo que significa que si adoptamos el camino de JavaScript para implementar Ajax habrá que escribir diferente código para los diferentes navegadores para asegurarse de que funciona en todos.

Pero afortunadamente JQuery simplifica el proceso implementando Ajax teniendo cuidado de las diferencias entre navegadores. Así que JQuery ofrece métodos tales como **`load()`**, **`$.get()`**, **`$.post()`**, etc, para implementar Ajax y que funcione correctamente en todos los navegadores.

En los próximos capítulos veremos los métodos para enviar y recibir datos usando HTTP GET y POST.

Load Ajax en JQuery

Método `load()`

El método `load()` carga datos de un servidor y devuelve el código HTML devuelto en un elemento seleccionado. Este método nos ofrece una forma muy simple de cargar datos de manera asíncrona desde un servidor web.

La sintaxis básica es la siguiente.

```
$(selector).load(URL,data, complete);
```

Veamos esto en un caso práctico. Vamos a crear un fichero HTML llamado "test-content.html" y lo salvamos. Y escribimos el siguiente código:

```
<h1>Demo simple de Ajax</h1>
<p id="hint">Esto es un simple ejemplo sobre como cargar elementos con Ajax.</p>
<p></p>
```

Ahora creamos otro fichero al que vamos a llamar "load-demo.html" y lo salvamos y le agregamos el siguiente código.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Cargar contenido externo gracias a Ajax</title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("#button").click(function(){
        $("#box").load("test-content.html");
    });
});
</script>
</head>
<body>
    <div id="box">
        <h2>Pulsa el botón para añadir nuevo contenido</h2>
    </div>
    <button type="button">Cargar contenido</button>
</body>
</html>
```

Además, la función de llamada puede tener tres parámetros diferentes:

- **responseTxt** – Contiene el contenido resultante si la petición es exitosa
- **statusTxt** – Contiene el estado de la petición tanto si la petición es exitosa o da error.
- **jqXHR** – Contiene el objeto XMLHttpRequest

Ejemplo:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $("button").click(function(){
        $("#box").load("/examples/html/test-content.html", function(responseTxt, statusTxt, jqXHR){
            if(statusTxt == "success"){
                alert("El nuevo contenido es cargado correctamente!");
            }
            if(statusTxt == "error"){
                alert("Error: " + jqXHR.status + " " + jqXHR.statusText);
            }
        });
    });
});
</script>
</head>
<body>
    <div id="box">
        <h2>Pulsa el botón para ver nuevo contenido</h2>
    </div>
    <button type="button">Carga el contenido</button>
</body>
</html>

```

Ejercicios

_Vamos a empezar con algo sencillo: _

1. Crear un documento HTML con un encabezado, dos párrafos y un botón. Al pulsar el botón mediante el selector \$("*") se debe ocultar todo. Método hide()
 - (a) En el mismo documento HTML anterior. Al pulsar el botón debe ocultarse dicho botón
 - (b) En el mismo documento HTML anterior. Suponemos que el encabezado y el primer párrafo tienen el atributo class="intro". Deberás ocultar al pulsar
 - (c) el botón, dicho párrafo.

2. Selecciona el primer elemento de la primera lista y oculta dicho elemento.
3. Crea un documento con un par de enlaces mediante el atributo href y oculta ambos enlaces.
4. Crea un documento con dos enlaces y dos botones. Uno se tiene que abrir en una nueva ventana (`target="blank"`) y el otro no. Cada botón debe ocultar uno de los enlaces en función de su atributo target.
5. Crear un documento con una tabla. Poner el fondo de las filas pares en rojo (usar el método `.css("background-color","red")`). A continuación poner el fondo de las filas impares en verde `.css("background-color","green")`
6. Crear un documento con un párrafo, el cuál se oculte al clicar dos veces sobre él
7. Crear un documento con un párrafo tal que aparezca un aviso `alert()` cuando se presiona el botón izquierdo del ratón.
8. Utiliza el método `hover()` para lanzar un mensaje cuando nos posicionamos sobre un párrafo y otro cuando salgamos de él.
9. Utiliza los métodos `focus()` y `blur()` para cambiar el color de dos cuadros de texto cuando posicionamos el foco y cuando lo retiramos.
10. Crea un documento con dos tag div, dentro de cada uno de los cuales debe haber un botón para esconder dichos tag.
11. Crea un documento con un botón, tal que al clickarlo como gestor de vento, agregue la opacidad de tres elementos div mediante el método `fadeIn()`. El primer `fadeIn()` ejecútalo sin parámetros, el segundo de manera lenta y el tercero en tres milisegundos.
12. Crea un documento con un botón, tal que al clickarlo nos quite la opacidad de tres elementos div mediante el método `fadeOut()`. El primer `fadeOut()` ejecútalo sin parámetros, el segundo de manera lenta y el tercero en tres milisegundos
13. Crea un documento con dos capas, tal que al clickar sobre la primera se despliegue hacia abajo la segunda mediante el método `slideDown()`.
14. Crear un programa, tal que al clickar en un botón se mueva un cuadrado 100 px a la derecha y si dentro del cuadrado pone HELLO aumente el tamaño de la letra.
15. En el ejercicio anterior incluir un botón para poder detener el efecto.
16. Mejora el ejercicio anterior añadiendo cuatro botones.
 - (a) Start. Comienza la animación

- (b) Stop. Parar la animación
 - (c) StopAll. Detiene todas las animaciones
 - (d) Stop but finish. Detiene las animaciones pero acabando primero la que se encuentra en ejecución
17. Crear un programa con un botón, tal que al hacer clic ponga las letras en rojo de un párrafo, lo oculte con un desplazamiento hacia arriba (slideUp) y lo visualice con un desplazamiento hacia abajo (slideDown).
 18. Mediante tres eventos asociados a tres botones, “SetText”, “SetHTML” y “SetValue” establecer el texto de un primer párrafo a “Hola Mundo ”, establecer el texto de un segundo párrafo a “Hola Mundo” en negrita y establecer el valor de un cuadro de texto a “Hola Mundo”.
 19. Cambia los atributos href y title de un enlace a Google, mediante un evento asociado a un botón. Ahora href=”<http://www.google.es/intl/es/earth/index.html>” y el título será “Google Earth”
 20. Cambia un enlace a <http://google.es> mediante un evento asociado a un botón y un callback, en el que el nuevo texto introducido sea /intl/es/earth/index.html
 21. Crear un documento con un párrafo, mediante un evento asociado a un botón debemos añadirle (indiferente principio o final) un párrafo creado con HTML, un párrafo creado con jQuery y un párrafo creado con el DOM de JavaScript.
 22. Crea una capa cuadrada (100 x 300 en amarillo) que contenga un par de párrafos. Borra dichos párrafos con el método empty() como gestor de un evento asociado a un botón.
 23. Añade una clase css a distintos elementos de HTML mediante un evento asociado a un botón y como gestor del evento el método addClass. Mediante otro evento asociado a un segundo botón elimina la clase css de los elementos utilizando como gestor el evento removeClass().
 24. Tenemos un documento con 4 párrafos cada uno con un color de fondo. mediante un evento asociado a un botón devuelve mediante una alerta el color de fondo del primer párrafo.
 25. En el mismo documento anterior, mediante un evento asociado a un botón. Cambia el color de fondo de los cuatro párrafos a amarillo y aumenta el tamaño de la letra
 26. Mediante un evento asociado a un botón obtener el ancho y alto de una capa cuadrada. Visualizarlo como texto dentro del elemento div.
 27. Obtener la altura y anchura del documento y de la ventana como respuesta a un evento asociado a un botón.
 28. Selecciona de una lista de 5 ciudades, la segunda, debes resaltarla con un color de fondo rojo

29. Crea una tabla de 8 filas, a continuación pon el fondo rojo a todas aquellas que estén por encima de la tercera (2) y pon el fondo azul a todas aquellas que estén por debajo de la tercera (2)
30. Selecciona todas las cabeceras de un documento y ponlas en rojo.
31. Selecciona de una tabla todas las casillas vacías y ponlas un color de fondo amarillo.
32. Selecciona todos los elementos con el atributo href acabado en .com y a continuación resalta dichos elementos en amarillo
33. Selecciona todos los elementos de tipo input de tu documento y ponlos con un color de fondo rojo
34. Realiza el mismo ejercicio que el anterior rodeando con un span en color rojo los elementos input de tipo radio y de tipo checkbox.
35. Crear un documento con un elemento div y dentro del div un par de párrafos, de tal manera que al hacer clic en un párrafo cambie el color de ambos párrafos.
36. Crear un documento con un párrafo, tal que al hacer clic aumente el tamaño de la letra. Sólo se producirá dicho efecto en dos ocasiones, luego el efecto debe desaparecer.
37. Crea un documento con dos botones, start y stop. Mediante el primero se debe producir la siguiente animación. Debe aparecer un elemento div cuadrado de 40 x 40, desplazarse a la izquierda 200 px, cambiar de color a azul y ocultarse desplazándose hacia arriba. **Solución**
38. Contar caracteres en un textarea.Crea un plugin para conseguir que un campo textarea de un formulario, informe en todo momento de los caracteres que ha escrito el usuario. Es decir, vamos a hacer un método del objeto jQuery que servirá para que cuente los caracteres en una capa de texto detrás de un textarea. **Solución**
39. Seguridad en una clave Crea un plugin que realice lo siguiente. Tendremos un formulario con un campo input password. Con jQuery mostraremos dinámicamente un mensaje al lado del campo con la fortaleza de la clave que haya escrita. A medida que el usuario cambie el contenido del campo, se actualizará el mensaje del lado, mostrando la fortaleza de la nueva clave. (<5 no segura, >5 y <8 medianamente segura, >8 segura) **Solución**
40. Campos adicionales Crear un plugin que permita añadir campos en un formulario. Tendremos un enlace en el formulario con el texto "Más campos" y al pulsarlo, simplemente se inyectará el código HTML para mostrar un campo nuevo en el formulario. **Solución**
41. Implementar la función tip sobre los elementos del documento.Realiza un sistema para crear elementos, tal que, al pasar el ratón sobre ellos, muestre un mensaje con una explicación contextual, lo que habitualmente se conoce como "tip" **Solución**

42. Crear un programa con jQuery para obtener un menú, tal que cada vez que el usuario introduce el puntero el item se establece de color blanco mediante un efecto persiana. **Solución**
43. Validar un formulario con jQuery. Se trata de un formulario en el que hay que introducir el nombre, mail, asunto y un mensaje. Si no se introduce alguno de los campos o si el mail es incorrecto nos aparecerá un tip informando de dicho asunto. En el enlace siguiente está el código del programa tan sólo falta la parte de jQuery. **Solución**
44. Hacer un efecto tipo acordeón con jQuery.
45. Hacer un Slidder con JQuery
46. Introduce en un documento el plugin jquery.snow.js para crear el efecto de nevando. Deberás implementar el tamaño mínimo del copo de nieve a 12 (minSize), el tamaño máximo a 18 (maxSize) , la frecuencia con la que cae la nieve a 400 (newOn) **Solución**
47. Realiza un programa en JQuery para que en tu documento aparezcan 8 fotos en orden aleatorio. **Solución**