**Nieves Borrero**

**Entrega de los códigos del manual jQuery. Parte 1, 2 y 3.**
**En un documento pdf responde a las siguientes preguntas:**

1. **En una línea, define qué es jQuery.**
   Un framework Javascript, es decir librerías de código que contienen procesos o rutinas ya listos para usar.

2. **Identifica la última versión jQuery**
   La 3.2.1

3. **Indica las diferencias entre la versión DEVELOPMENT, PRODUCTION y SLIM.**
   La versión production es la adecuada para páginas web en producción, puesto que está minimizada y ocupa menos espacio, con lo que la carga de nuestro sitio será más rápida.
   La versión development está comentada.
   La versión SLIM elimina algunas funciones.
4. **Indica la línea donde introduces todo el código de la librería jQuery.**
   Dentro de una etiqueta script en el head de mi html.

5. **Indica qué es el jQuery CDN.**
   Significa Content Delivery Network, que se traduciría como red de entrega de contenido. Se trata de un servicio que nos permite incluir las librerías de código de jQuery desde los servidores de algunas importantes empresas. En vez de enlazar con los scripts de jquery alojados en nuestro sitio web, linkamos directamente con una URL de otro dominio que los aloja por nosotros.

6. **Indica brevemente y con tus palabras las ventajas del CDN de jQuery.**
   -Mayor velocidad de entrega: Los servicios CDN están ofrecidos por grandes empresas, con replicación de servidores y diversas localizaciones de entrega a lo largo del mundo, de manera que puede enviar el script jQuery más rápido y distribuirlo desde una localización probablemente más cercana a la red del  tus clientes.
   -Cacheado probable:
   Es muy probable que la persona que te visita ya haya cacheado el script jQuery, tras la visita a otra página web que esté usando también el CDN de alguna de estas empresas. Por ello quizás no tenga ni que esperar a que descargar el framework y utilice la copia que ya tiene en la caché del navegador.

7. **Indica la línea donde introduces las últimas versiones de al menos dos jQuery CDN.**
   <script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>

8. **Indica cómo jquery ejecuta un código cuando el árbol DOM está totalmente cargado. Indica el equivalente en javaScript.**
   $(document).ready() -> equivalente en js: window.onload()
   Pero con window.onload() sólo se ejecutará cuando el navegador haya descargado completamente TODOS los elementos de la página, lo que puede tardar mucho y no es

necesario, basta con ejecutarlo cuando el navegador ha recibido el código HTML completo y lo ha procesado, que es como lo hace ready.

9. **Función $ o función jQuery. Indica brevemente los argumentos que puedes enviarle. Añádele a la explicación un breve código de ejemplo (distinto al del manual)**
   - Un selector. Parámetro obligatorio.

```
let divContenido = $("#container"); // Seleccionamos el elemento con id container
divContenido.css(background-color", "gray"); // Hacemos algo con él
```

   … y un contexto opcional

```
Let parrafos= $("p",document.article[0]); // Seleccionamos todos los párrafos de
                                          dentro del primer article de la página
parrafos.css("color", "blue"); // Cambiamos el color de la letra.
```

   Para seleccionar elementos o grupos de elementos de una página web …

   - Un html. Para crear directamente los elementos.

```
Let elementos = $("<h1>Creando un h1</h1><p>y un párrafo en tiempo de
ejecución</p>");
elementos.appendTo("body");
```

   - Una función. Que viene a ser una función callback que se invoca automáticamente cuando el DOM está listo.

```
$(document).ready(function(){
        $("enlace").click(function(evento){
                event.preventDefault();
                $("#divOculto").css("display", "inline");
});
```

10. **Indica cómo puedes reemplazar el clásico $(document).ready(){...} con jQuery**
    $(window).load()

11. **En una línea, explica qué hace el método each() de jQuery. Explica qué es la iteración implícita.**
    Itera sobre todos los elementos del DOM que se hayan seleccionado realizando la función que recibe como parámetro por cada uno de los elementos a los que accedemos mediante la palabra this.

    La iteración implícita hace referencia al recorrido que hacen la mayoría de los métodos que devuelven un objeto jQuery recorriendo el conjunto de elementos de la colección jQuery. Cuando esto ocurre, a menudo no es necesario iterar explícitamente con el método .each ()

    Ejemplo:

    ( "li" ).addClass( "lista" );

12. **Indica el argumento que ha de enviársele al método each().**

Una función.

**13. Englobado en el contexto del each:**

1. **Explica la utilidad de la palabra reservada this.**
   Hace referencia al elemento sobre el que iteramos.
2. **Indica cómo se utiliza el índice de la iteración.**
   A la función each, le pasas como parámetro una variable que es el índice.

```
$(document).ready(function(){
	$("p").each(function(i){
		if(i%2==0){
			$(this).css("background-color", "#eee");
		}else{
			$(this).css("background-color", "#ccc");
		}
	});
});
```

3. **Explica la utilidad de return false.**
   Permite salir de la función comportándose como un break, mientras que return true, es como un continue.
4. **Indica la diferencia entre return true y no ponerlo. Explícalo mediante un trozo de código.**
   La diferencia está en que al poner return true salta esta iteración sin ejecutar el código restante y continua iterando sobre los elementos siguientes. El no ponerlo hará que siga ejecutando el código dentro de esa iteración.

```
$(document).ready(function(){
	$("div").each(function(i){
		elemento = $(this);
		if(elemento.html() == "white")
			return true; // Sale del código y continúa iterando sobre los
					siguientes elementos, de manera que no pinta
					las letras de blanco.
		if(elemento.html() == "nada")
			return false;
		elemento.css("color", elemento.html());
	});
});
```

   Sin return true pintaría también ese texto de blanco.

14. **Indica las diferencias y semejanzas entre el método size() y la propiedad length. Indica las ventajas e inconvenientes de utilizar uno u otra.**
    Ambos sirven para obtener el número de elementos seleccionados con la función jQuery.

Size simplemente devuelve el número de elementos que hay en el objeto (pero está deprecated) y length almacena directamente este valor, por lo que es más rápido y más aconsejable que calcular los elementos seleccionados con el método size()

15. **Indica qué hace el método data().**

sirve tanto para guardar un dato en un elemento como para consultarlo. Según el número de parámetros que reciba, realiza una u otra acción. 1→guarda, 2->consulta

16. **Tipos de datos admitidos por data()**

cualquier tipo de variable, cadena de texto o numérica, un array o incluso un objeto Javascript o jQuery.

17. **Indica qué significa que data() almacena valores por referencia.**

Que no se harían copias independientes del objeto guardado, sino que se asigna por referencia, distintas referencias al mismo objeto.

18. **Cuántos objetos se crean si data() opera sobre un conjunto de elementos.**

Uno solo.

19. **Indica qué hace el método removeData().**

Elimina el dato almacenado en el elemento

20. **Identifica con tu propio código (en una línea a ser posible) los distintos tipos de selectores. Indica cómo se recogen en una variable.**

-Por etiquetas $(“p”);

-Por identificador $(“#id);

-Por clase $(“.clase”);

-Por varias clases $(“.claseUno .claseDos”);

-Todos los elementos $(“*”);

-Concatenando varios selectores $(p, id, .clase);