Elisa Navarro Zuara jQuery. Parte I, II y III

1. En una línea, define qué es jQuery.

   Es un framework de Javascript.

2. Identifica la última versión jQuery.

   jQuery 2.2.0

3. Indica las diferencias entre la versión DEVELOPMENT y PRODUCTION.

   La versión PRODUCTION contiene el código de forma minificada y ocupa menos espacio, con lo que la carga de nuestro sitio será más rápida aunque el código no es legible. La versión DEVELOPMENT el código está sin comprimir, con lo que ocupa más espacio, pero es legible.

4. Indica la línea donde introduces todo el código de la librería jQuery.

   <script type="text/javascript" src="js/jquery-2.2.0.min.js"></script>

5. Indica qué es el jQuery CDN.

   jQuery CDN (Content Delivery Network) es un servicio que nos permite incluir las librerías de código de jQuery desde los servidores de algunas importantes empresas.

6. Indica brevemente y con tus palabras las ventajas e inconvenientes del CDN de jQuery.

   Ventajas
   - Mayor velocidad de carga de la página, ya que los servicios CDN están ofrecidos por grandes empresas.
   - Almacenamiento del script jQuery en la caché del navegador.

   Inconvenientes
   - Necesitamos estar conectados a Internet para acceder al CDN.
   - Menor control sobre el script, ya que no puede ser modificado si es necesario.

7. Indica la línea donde introduces las últimas versiones de al menos dos jQuery CDN.

   <script type="text/javascript" src="https://code.jquery.com/jquery-2.2.0.min.js"></script>
   <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.4.min.js"></script>

8. Indica cómo jQuery ejecuta un código cuando el árbol DOM está totalmente cargado. Indica el equivalente en JavaScript.

   jQuery
   ```
   $(document).ready(function() {
           // Código
   });
   ```

   JavaScript
   ```
   window.onload = function() {
           // Código
   }
   ```

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).

   Los argumentos que se pueden enviar son:
   - Un selector y un contexto.
   - Un string con un HTML.
   - Un elemento o una jerarquía de elementos del DOM.
   - Una función.

   Código de ejemplo:

   ```
   $(function() {
           $("#elemento1").css("color", "#000000");
   });
   ```

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

    ```
    $(function() {
            // Código
    });
    ```

11. En una línea, explica qué hace el método each() de jQuery. Explica qué es la iteración implícita.

    Realiza una iteración por todos los elementos del DOM que se hayan seleccionado.
    La iteración implícita nos evita tener que estar programando bucles de código para buscar todos elementos que cumplen el criterio dado.

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

    El método each() recibe una función.

13. Englobado en el contexto del each:

1. Explica la utilidad de la palabra reservada this.

   Con la palabra reservada "this" tenemos acceso al elemento actual.

2. Indica cómo se utiliza el índice de la iteración.

```
$("div").each(function(i) {
        if (i%2==0) {
            $(this).css("color", "#FFFFFF");
        } else {
            $(this).css("color", "#000000");
        }
});
```

3. Explica la utilidad de return false.

   Si la función devuelve "false", se consigue detener por completo el proceso de iteraciones de each(). Esto es como si hiciéramos el típico "break".

4. Indica la diferencia entre return true y no ponerlo. Explícalo mediante un trozo de código.

   La diferencia es que al poner return true se consigue pasar directamente a la próxima iteración del bucle sin seguir ejecutando el código de la iteración actual. Si no se pone return true se ejecutará todo el código de la iteración.

   Código de ejemplo:

```
<div>elem1</div>
<div>elem2</div>
<div>elem3</div>

$(function() {
        $("div").each(function(i) {
            if ($(this).html == "elem2") {
                    console.log("Bienvenido al elemento 2");
                    return true;
            }
            alert("Iteración por los elementos");
        });
});
```

   Resultado con return true:

   - Iteración por los elementos
   - Bienvenido al elemento 2
   - Iteración por los elementos

Resultado sin return true:

- Iteración por los elementos
- Bienvenido al elemento 2
- Iteración por los elementos
- Iteración por los elementos

14. Indica las diferencias y semejanzas entre el método size() y la propiedad length. Indica las ventajas e inconvenientes de utilizar uno u otra.

    El método size() y la propiedad length sirven para obtener el número de elementos seleccionados con la función jQuery. Aunque la propiedad length es más rápida ya que almacena directamente el valor.

15. Indica qué hace el método data().

    Este método del objeto jQuery sirve tanto para guardar un dato en un elemento como para consultarlo. Según el número de parámetros que reciba, realiza una u otra acción.

    - data(nombre): devuelve el valor que haya en el dato cuyo nombre se pasa por parámetro.
    - data(nombre, valor): almacena un dato, cuyo nombre recibe en el primer parámetro, con el valor que recibe en el segundo parámetro.

16. Tipos de datos admitidos por data().

    Admite cualquier tipo de dato.

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

    Esto quiere decir que no se harían copias independientes del objeto a guardar, sino que permanecería tal cual y lo que se asignaría como dato es una referencia a ese único objeto.

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

    Se crea la misma cantidad de objetos que el número de elementos.

19. Indica qué hace el método removeData(). ¿Está sobreescrito?

    Este método sirve para eliminar un dato de un elemento. No está sobreescrito.

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

    $("div"), $("#idelemento"), $(".miclase"), $(".clase1.clase2"), $("*")

    var elementos = $("div");