| Mostrando Marca_Txurdinaga_Color_H_SUP_01_01.jpg | Curso / *Kurtsoa* | Fecha / *Data* | Nivel / *Maila* | Eval. /*Ebal.* |
| --- | --- | --- | --- | --- |
| 2023-24 | 2023/11/09 | 2º | 1ª |
| Módulo / *Modulua* | *Kodea / Código* | *U.Didak /Unid didác* | *tipo* | Calificación/*Kalifikazioa* |
| DWC | 2DW3 | 1, 2, 3, 4, 5, 6 | Prozedurazkoa |  |
| Nombre Alumno/a / Ikaslearen izena |  | | | |

* *Para la resolución de estos problemas se valorará positivamente: la claridad del código, la separación del código Javascript del HTML, el uso de convenciones para nombrar funciones y variables, y el uso de comentarios en el código, así como la corrección de los resultados obtenidos.*
* *El examen vale 10 puntos, si la propuesta del examen se resuelve de forma correcta se le aplica el valor especificado en cada apartado.*
* *Utiliza los ficheros.js que se te adjuntan para escribir cada parte del examen.*
* *Cuando terminéis subiréis un sólo archivo comprimido, con el nombre EVA1\_Apellidos\_Nombre.*

**El punto limpio de tu localidad te ha solicitado un programa que lleve un control de los residuos que se van depositando en los diferentes contenedores y qué personas los han entregado.**

Para ello dispones de los siguientes archivos:

* Un HTML con dos partes: la primera tiene cinco campos de texto con la información de los kg que contiene cada contenedor.
* En la segunda hay un formulario que registra las descargas, y controla: quién hace la descarga, cuántos kg deposita y en qué contenedor.

IMPORTANTE: no se puede modificar ningún dato del HTML.

Debes crear un programa completo que permita realizar la gestión, y para ello te proponemos que lo vayas haciendo paso a paso:

**1. CLASES Y OBJETOS (descarga.js)**

Crea una clase Descarga con su constructor, atributos, métodos get y set y un método mostrar.

Los getters y setters deben estar creados como recomiendan los estándares, y no crear tus propios nombres de métodos para este caso.

Dispondrá de los siguientes elementos:

* Color: representa el color del contenedor.
* Residuo: indica qué tipo de residuo se puede depositar.
* Kg: representa el número de kg que se depositan en la descarga.
* Nombre: almacena el nombre de la persona que ha llevado los residuos al punto limpio.

El método mostrar, por su parte, deberá devolver una frase como la siguiente:

“<Nombre> ha depositado <kg> kg del residuo <residuo> en el contenedor <color>”.

| **Total ejercicio** | **1,5** |
| --- | --- |
| 1-El objeto ha sido definido correctamente | 0,25 |
| 2-Todos los getters y setters siguen los estándares | 0,75 |
| 3-El método constructor ha sido definido correctamente | 0,25 |
| 4-El método mostrar ha sido definido correctamente | 0,25 |

**2. ARRAYS (array.js)**

En el archivo array.js deberás crear dos métodos con las siguientes especificaciones:

* **addArray**: recibirá dos parámetros: un array de objetos Descarga y un elemento. Añadirá el elemento al array por el principio. A continuación, hará una llamada al método mostrarArray.
* **mostrarArray**: recibirá como parámetro el array de objetos Descarga. A continuación, recorrerá dicho array y mostrará los elementos que contiene por consola utilizando para ello el método mostrar definido en el objeto.

| **Total ejercicio** | **1,25** |
| --- | --- |
| 1-La función addArray añade el elemento al array | 0,5 |
| 2-La función mostrarArray muestra todos los elementos del array en consola | 0,75 |

**3. FORMULARIO (script.js, validar.js y pintar.js)**

Fíjate bien en el ***select*** del formulario. Podrás observar que cada ***option*** tiene el siguiente formato:

<option value=”Amarillo”>Envases</option>

Por lo que en función de lo que elija el usuario, podrás extraer toda la información necesaria para crear objetos.

**Nota**: para extraer cuál es la opción seleccionada por el usuario, puedes hacerlo de la siguiente manera a partir del elemento:

elementoSelect.options[elementoSelect.selectedIndex]

Con esto obtendrás el option seleccionado. A partir de ahí podrás extraer el texto (text) o el valor (value) en función de lo que necesites.

Importante: fíjate en las mayúsculas.

Al pulsar el botón no se deberá enviar el formulario, sino que se realizarán las siguientes operaciones que pondrán en funcionamiento todo lo que has creado hasta ahora (objetos, array, etc.)

1. **Validar**: Las validaciones se realizarán mediante Javascript no mediante HTML.
   1. Comprueba que has introducido un número positivo del 1 al 100 en el campo Kg.
   2. El nombre estará formado por letras mayúsculas o minúsculas y espacios, como mínimo 3 letras y como máximo 20. Para esta segunda comprobación deberás utilizar una expresión regular.
   3. Los errores que se produzcan se mostrarán en un alert. Si hay un solo error, se mostrará un error; si hay dos errores, se mostrarán los dos a la vez en el mismo alert.
   4. En caso de que la validación sea correcta, se continuará con el resto de operaciones; de lo contrario se parará el programa.
   5. En ningún caso se enviará el formulario a otra página.
2. Crear un nuevo objeto de tipo Descarga que tendrá las siguientes propiedades:
   1. Color: será el valor del input seleccionado.
   2. Residuo: será el texto del input seleccionado.
   3. Kg: será el valor introducido en el input Kg.
   4. Nombre: será el nombre introducido en el input Nombre.
3. Crea un párrafo con el siguiente texto en el div “descargas”:

“<Nombre> ha descargado <kg> kg de residuo <residuo> en el contenedor <color> el día <fecha en formato dd/mm/yyy>”

Para ello deberás utilizar el objeto recién creado y los métodos correspondientes, es decir, no debes utilizar los valores extraídos del formulario directamente.

1. Crea una función llamada actualizarKg que actualice el valor del input correspondiente de manera que si, por ejemplo, el usuario ha elegido 10 kg en el contenedor marrón y ya había un valor de 20, se actualice a 30.   
   Se valorará positivamente la programación eficiente en este apartado de manera que no se utilicen condicionales para elegir el campo a modificar, sino que se elija automáticamente.
2. Utiliza los métodos del archivo array.js para añadir un elemento nuevo al array y que se muestre por consola.
3. Por último, verás que dispones de una función llamada pintarColor almacenada en el archivo pintar.js. Esta función recibe el elemento párrafo que acabas de crear y le añade una clase en función del color del mismo.  
   Debes pensar el modo de hacerlo de una manera mucho más eficiente que no dependa del color que le pasemos por parámetro sino que utilice los datos de los que ya dispones.

| **Total ejercicio** | **7,25** |
| --- | --- |
| 1-El formulario no se envía | 0,25 |
| 2-Validación de kg | 0,25 |
| 3-Validación de nombre | 0,25 |
| 4-Validación de nombre mediante expresión regular | 1,00 |
| 5-Todos los errores se muestran en un solo alert | 0,75 |
| 6-Si todo es correcto se genera un nuevo objeto Descarga | 0,50 |
| 7-Creación del párrafo con los datos del objeto creado | 1,50 |
| 8-Actualiza los kg del contenedor correspondiente | 0,75 |
| 9-Añade el objeto al array | 0,50 |
| 10-Muestra los elementos del array por consola | 0,50 |
| 11-Optimiza pintar.js, elimina el código propuesto y da una solución mejorada | 1,00 |
|  |  |