Sesión 09

Proporcionar una clase que permita crear objetos que se incrustarán como elementos interactivos en un documento (X)HTML5 de forma similar a como se indica en las transparencias del tema (pág. 76 a 80). Para esta práctica los elementos a incrustar serán cronómetros y ya se proporciona un documento, `chrono.html`, que incluye el código necesario para añadir dos cronómetros en una sección del documento.

Todas las clases necesarias, variables y funciones auxiliares se declararán en un espacio de nombres independiente, para lo cual las declaraciones se realizarán dentro una función que se autoinvoque. Esta función retornará lo que se requiera exportar, a priori, la función constructora de la clase `State` que mantiene el estado de los cronómetros, de forma que el esquema del código será el siguiente:

```
// namespace. Función que se autoinvoca
const Chrono = (function () {
   // Clase opcional de configuración (singleton)
   // que para esta práctica es innecesaria

   // Clases auxiliares
   // tiempo (campos: h, m, s y d) del cronómetro
   class Time {...}
   // eventos (campos: start, stop y reset de valor
   // cierto o falso) según el estado del cronómetro
   class Events {...}

   // Clase principal. Estado del cronómetro
   class State {...}
   // Subárbol de la Interfaz del cronómetro
   class View {...}

   // Declaración de funciones y variables auxiliares

   return State;
})();
```

En el esquema del código se propone utilizar dos clases auxiliares una para el tiempo del cronómetro (`Time`) y otra para registrar y manejar los eventos que se pueden producir al intervenir el usuario (`Events`).

La clase `Time` mantendrá las horas, minutos, segundos y décimas de segundo del cronómetro, y dispondrá de los métodos siguientes:

- `reset()`. Restablece el tiempo a `0`.
- `incr()`. Incrementa el tiempo en una décima de segundo
- `toString()`. Retorna el tiempo como la cadena `hh:mm:ss.d`. Para facilitar la obtención de ésta se proporciona la función auxiliar `int2str()`.

La clase `Events` mantendrá tres campos booleanos, `start`, `stop` y `reset`. Estos campos tendrán el valor `true` siempre que el usuario pueda actuar sobre el cronómetro para realizar la acción correspondiente: arrancar, parar o reiniciar el cronómetro, respectivamente. Para facilitar el tratamiento repetitivo de los eventos, así como de los botones actuables que los provocan, se ha definido el array auxiliar `actions`.

La clase `State` mantiene el estado del cronómetro y las operaciones que permiten manipularlo:

- `time`. Objeto de clase `Time` que mantiene el tiempo del cronómetro
- `events`. Objeto de clase `Events` que mantiene la información de las posibles actuaciones del usuario sobre el cronómetro (arrancarlo, pararlo o reiniciarlo).
- `view`. Objeto de clase `View`, la interfaz del cronómetro.
- `strObject`. Un identificador único del cronómetro. Para obtener el identificador se proporciona la función auxiliar `idChrono()`.
- `start()`. Arranca o continua el tiempo cronometrado.
- `stop()`. Para el tiempo cronometrado.
- `reset()`. Reinicia el tiempo del cronómetro.
- `incr_time()`. Incrementa el tiempo cronometrado en una décima de segundo.
- `node()`. Nodo raíz del subárbol que establece la interfaz del cronómetro.

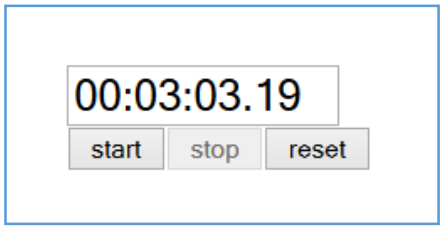

Fig. 1. Interfaz del cronómetro

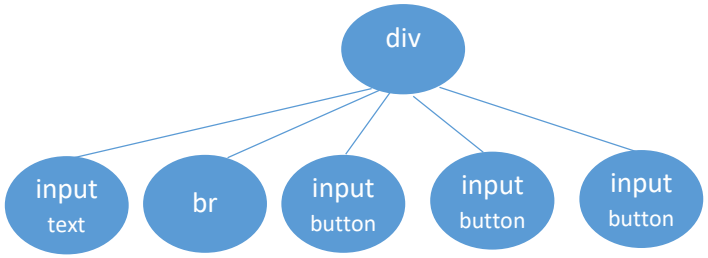

Fig. 2. Subárbol de nodos de la intefaz

La clase `View` mantiene la interfaz del cronómetro, para poder acceder al estado del cronómetro como mínimo se requiere conocer éste (su referencia). La interfaz es un subárbol de elementos HTML que permite presentar la información requerida del cronómetro (su tiempo) y actuar sobre éste, por ejemplo, mediante botones actuables (ver figuras 1 y 2). Estos elementos, que esquemáticamente se indican a continuación, se deben generar de forma dinámica.

La operación principal de la interfaz, que además podría haber sido la única disponible, es `show()`. Esta operación es la encargada de actualizar la vista de la interfaz cuando se requiera: cuando cambia el tiempo (cada décima de segundo) o se produce un cambio de estado del cronómetro por intervención del usuario.

Esquema de la interfaz HTML del cronómetro

```
<div id="chrono_0" class="chrono">
   <input type="text" .../><br/>
   <input type="button" .../>
   <input type="button" .../>
   <input type="button" .../>
</div>
```

Si bien no es imprescindible, ya que es posible localizar y acceder a cualquier nodo del árbol del documento (X)HTML5 mediante las operaciones del DOM, se facilitará el acceso en tiempo de ejecución al nodo raíz de la interfaz y a los nodos de ésta que se modifican (todos menos el elemento `br`) haciendo que éstos sean parte del estado de la interfaz.

Métodos del DOM

`document.createElementNS(xmlns, tag)`. Crea un nodo elemento de etiqueta `tag` en el espacio de nombre especificado.

Los nodos creados sólo se muestran cuando se incorporan al árbol del documento, por ejemplo, mediante el método `parent.appendChild(node)`, que añade el nodo especificado como último hijo del nodo receptor.

Para añadir, acceder o quitar atributos de los elementos se pueden utilizar los métodos de elementos: `setAttribute`, `getAttibute` y `removeAttribute`, respectivamente.

Puedes ver ejemplos de uso en la URL:

http://www-sew.dyndns.org/aii/ejemplos.php?id_codigo=4&codigo=DOM

Actualización del tiempo del cronómetro

Para actualizar el tiempo del cronómetro es necesario utilizar el método `window.setInterval()` que permite ejecutar una función a intervalos regulares expresados en milisegundos. Puedes ver la sintaxis y funcionamiento en la URL:

http://www-sew.dyndns.org/aii/ejemplos/ejemplo.php?id_ejemplo=243&volver=8DOM-HTML55

El método retorna un identificador del temporizador creado y éste se puede pasar como argumento al método `window.clearInterval()` para eliminar, cuando se requiera, el temporizador puesto mediante `window.setInterval()`.

Observa que una posible forma de invocar el método `setInterval` para el primer cronómetro, **pero que tiene graves problemas**, es la siguiente:

```
window.setInterval("a.incr_time()", 100)
```

ya que `a` es la variable a la que se asigna dicho cronómetro en el *script* de la cabecera del documento (X)HTML5.

De forma similar, se podría utilizar la variable `a` en el atributo `onclick` para lanzar la acción correspondiente cuando se pulsa un botón, por ejemplo para el botón de arranque el elemento `input` debería incluir: `onclick="a.start()"`.

Con este código todo iría bien, pero como ya se indicaba tiene un grave problema, no hay forma de saber a priori cuál va a ser la variable a la que se va asignar un cronómetro. Hay que buscar por tanto una forma alternativa de referenciar éste y ésta tampoco puede ser `this`.

Observa que en la invocación `window.setInterval("this.incr_time()", 100)` el objeto `this` es `window` y no el cronómetro.

Una alternativa, es generar un identificador único para cada cronómetro y tener un objeto de clase (campo estático) en el que se asocie este identificador con la referencia `this` del cronómetro cuando éste se crea (en el constructor). De esta forma, y así está hecho en la clase `State`, se puede escribir un método estático (o de clase) que devuelva la referencia del cronómetro dado su identificador (método `instance()`) independientemente de la variable a la que finalmente se asigne el objeto cronómetro.