Aplicaciones telemáticas, examen práctico, 23/05/2024
Grado en Ingeniería Telemática
Universidad Rey Juan Carlos

Instrucciones

1. Emplea lo visto en clase, y solo lo visto en clase (que puedes consultar en las transparencias). No será válido ni JavaScript más antiguo ni más avanzado.
2. En los ficheros `~/figuras.html` y `~/figuras.js` encontrarás tu práctica 4.8. Tendrás que modificar el html para indicar que el script ya no es `tablaRandom.js` sino `figuras.js`. Encontrarás tus imágenes de la práctica 4 en `~/images`. Si tu código está bien hecho, esto no necesitará ninguna adaptación.
3. Cuando acabes, ejecuta la orden `comprime_at`. Esto creará el fichero `~/Escritorio/at.zip` con todo tu trabajo, para que lo entregues mediante el aula virtual.

Ejercicio 1. (3.5 puntos)

Modifica el fichero `~/clases.js` según la siguiente especificación:

El DOM nos ofrece los métodos *add()*, *remove()*, *contains()* y *toggle()* del objeto *classList* para facilitar la manipulación de las clases de un objeto. Pero estos métodos no son imprescindibles, cualquier programador debería poder implementarlos sin mucha dificultad. Demuestra que eres programador de JavaScript haciendo precisamente eso: unas funciones *add()*, *remove()*, *contains()* y *toggle()* que, a partir de una cadena conteniendo palabras que interpretaremos como clases, separadas por espacios:

1. La función *add(clases, clase)* devuelve una cadena conteniendo las clases originales, más la nueva clase. Si ya existía, no la añade de nuevo.
2. La función *remove(clases, clase)* devuelve una cadena conteniendo la cadena original, pero eliminando la clase. Si la clase no existe, no sucederá nada y las clases devueltas serán idénticas a las de entrada.
3. La función *contains(clases, clase)* devolverá *true* o *false* según la clase esté contenida o no en las clases.
4. La función *toggle(clases, clase)*, en caso de que la clase esté contenida, la eliminará. Y en otro caso, la añadirá. Naturalmente, esta función podrá llamar a las anteriores.

A continuación tienes un ejemplo de cómo deberían comportarse tus funciones:

```
let ejemplo = "aaa bbb";
ejemplo = add(ejemplo,"ccc");
console.log(ejemplo); // aaa bbb ccc
ejemplo = add(ejemplo,"ccc");
console.log(ejemplo); // aaa bbb ccc

console.log(contains(ejemplo, "bbb"))  // true
console.log(contains(ejemplo, "bb"))  // false

ejemplo = remove(ejemplo, "bbb")
```

```
console.log(ejemplo);  // aaa ccc

ejemplo = remove(ejemplo, "bb")
console.log(ejemplo);  // aaa ccc

ejemplo = toggle(ejemplo,"ccc");
console.log(ejemplo);  // aaa
ejemplo = toggle(ejemplo,"ccc");
console.log(ejemplo); // "aaa ccc"
```

Observaciones

- Para eliminar una clase de la lista de clases, alguien podría pensar que basta con reemplazar la subcadena que contiene la clase por la subcadena vacía. Pero esta esa solución no es válida. Por ejemplo, para eliminar `bb` de las clases `aa bb cc`, reemplazar `bb` por la cadena vacía. En este ejemplo, serviría. Pero dadas las clases `aa aaa`, si intentamos eliminar la clase `aa` de esta forma, en vez de devolver `aaa`, devolvería `a`, lo que es un error. No uses este método.
- Para buscar una clase en la lista de clases, alguien podría pensar que basta con buscar una subcadena en la cadena con las clases. Pero esta esa solución no es válida. Por ejemplo, la subcadena `bb` está contenida en la cadena `aaa bbb ccc`. Pero la clase `bb` no pertenece a esta lista de clases. No uses este método.
- Sugerencia: para evitar los problemas anteriores, convierte la cadena en una array de palabras, opera con este array y luego vuelve a convertirlo en cadena. Para que esto te resulte más sencillo, ya tienes disponible el código fuente de la función *lista_a_cadena*.

Solución

```
'use strict'

function contains(clases, clase){
    let encontrado = false;
    let clases_como_lista = clases.split(" ");

    for (let c of clases_como_lista){
        if (c === clase) {
            encontrado = true;
        }
    }
    return encontrado;
}

function add(clases, clase){
    if (contains(clases, clase) === false){
        clases = clases +" "+ clase;
    }
    return clases
}
```

```
function remove(clases, clase){
    let clases_como_lista = clases.split(" ");
    let nueva_lista = []
    for (let c of clases_como_lista){
        if (c !== clase){
            nueva_lista.push(c);
        }
    }
    let rval = lista_a_cadena(nueva_lista);
    return rval;
}


function toggle(clases, clase){
    if (contains(clases,clase)){
        clases = remove(clases,clase);
    } else {
        clases = add(clases,clase);
    }

    return clases;
}


function lista_a_cadena(lista){
    // Recibe un array de clases. P.e. ['aaa', 'bbb']
    // Devuelve una cadena con las clases, separadas por espacios.
    // P.e. 'aaa bbb'

    let rval = ""
    for (let c of lista){
        rval = rval + c + " ";
    }

    rval = rval.trim(); // Borramos el último espacio
    return rval;
}


let ejemplo = "aaa bbb";
ejemplo = add(ejemplo,"ccc");
console.log(ejemplo); // aaa bbb ccc
ejemplo = add(ejemplo,"ccc");
console.log(ejemplo); // aaa bbb ccc

console.log(contains(ejemplo, "bbb"))  // true
console.log(contains(ejemplo, "bb"))  // false

ejemplo = remove(ejemplo, "bbb")
console.log(ejemplo);  // aaa ccc

ejemplo = remove(ejemplo, "bb")
console.log(ejemplo);  // aaa ccc

ejemplo = toggle(ejemplo,"ccc");
console.log(ejemplo);  // aaa
ejemplo = toggle(ejemplo,"ccc");
console.log(ejemplo); // "aaa ccc"
```

Ejercicio 2. (6.5 puntos)

Modifica los ficheros `~/figuras.html`
y `~/figuras.js` según esta especificación:

- En modo gráfico, el ejercicio tiene que seguir funcionando como hasta ahora.
- Hasta ahora, el usuario podía marcar o desmarcar una figura haciendo *clic* sobre ella. Esto provocaba que o bien se mostrase la figura normalmente, o bien una imagen de reverso o similar. Cuando el usuario marcaba una carta o similar en modo texto, no sucedía nada.
- Modifica tu práctica para que las figuras en modo texto tengan un comportamiento similar al de las figuras en modo gráfico: al hacer clic sobre una, su texto será reemplazado por una serie de asteriscos, tantos como caracteres hubiera originalmente. Al volver a hacer clic, el texto volverá a su estado original. Las figuras en modo texto seleccionadas aparecerán en la misma estructura de datos que las figuras en modo gráfico. En esta estructura de datos no se distinguirán las figuras en modo texto o en modo gráfico.

Ejemplo: al hacer clic sobre *3 de corazones* el texto cambiará a *********** y el 3 de corazones aparecerá en la estructura que almacena las figuras seleccionadas. Al hacer clic sobre ***********, el texto volverá a ser *3 de corazones* y esta figura desaparecerá de la estructura de figuras seleccionadas.