Examen

Final febrero 2018

Aplicaciones Web
Año 2017/2018
Grado en Ing. Software
Fdi - UCM

Instrucciones generales:

- La duración del examen es de **tres horas**. Su peso en la calificación de la asignatura es del 60 %. Para obtener la calificación de APTO es necesaria la nota mínima de 4 sobre 10.
- No se permite ningún tipo de material sobre la mesa, salvo un bolígrafo y este enunciado.
- El examen consta de cuatro ejercicios que se realizan en el ordenador. Para su realización se proporciona un proyecto plantilla. Tras el enunciado de cada ejercicio se indica qué ficheros hay que modificar o añadir.

▷ 1. HTML y CSS

[1 pt] Supongamos el siguiente fragmento de código HTML que define la estructura de la cabecera y de la barra de navegación de una página web:

```
<div class="cabecera">
  <h1>¡Bienvenido!</h1>
  <p>No estás conectado</p>
</div>
<nav>
  <ul>
    <li><a href="...">Identificarse</a></li>
    <li><a href="...">Crear usuario</a></li>
    <li><a href="...">Información</a></li>
    <li><a href="...">Contacto</a></li>
  </ul>
</nav>
```

Modifica la hoja de estilo CSS para que la página tenga un aspecto como el mostrado en la Figura 1 (derecha). Ten en cuenta que todos los textos han de aparecer verticalmente centrados en sus respectivos contenedores.

Instrucciones para el ejercicio 1

- Modifica el fichero `public/css/ej1.css` para realizar este ejercicio. Los únicos atributos a insertar son los relacionados con *Flexbox*.
- Puedes comprobar el resultado abriendo `public/ej1.html` en el navegador. Puedes modificar el código HTML si quieres, aunque no es necesario para la realización de este ejercicio.

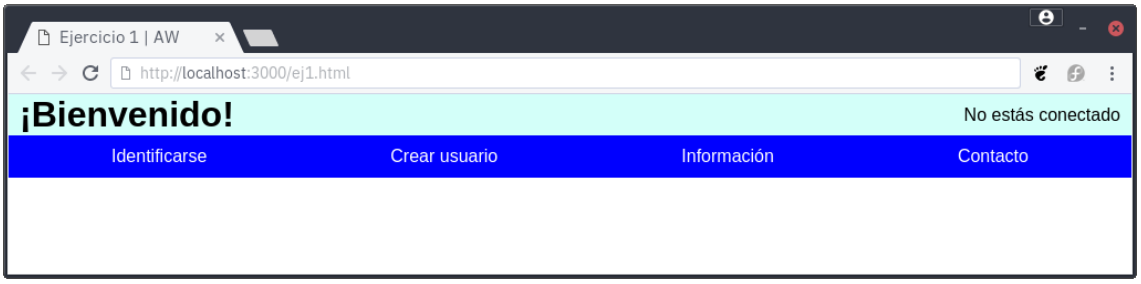


Figura 1: Cabecera y barra de navegación de un sitio web.

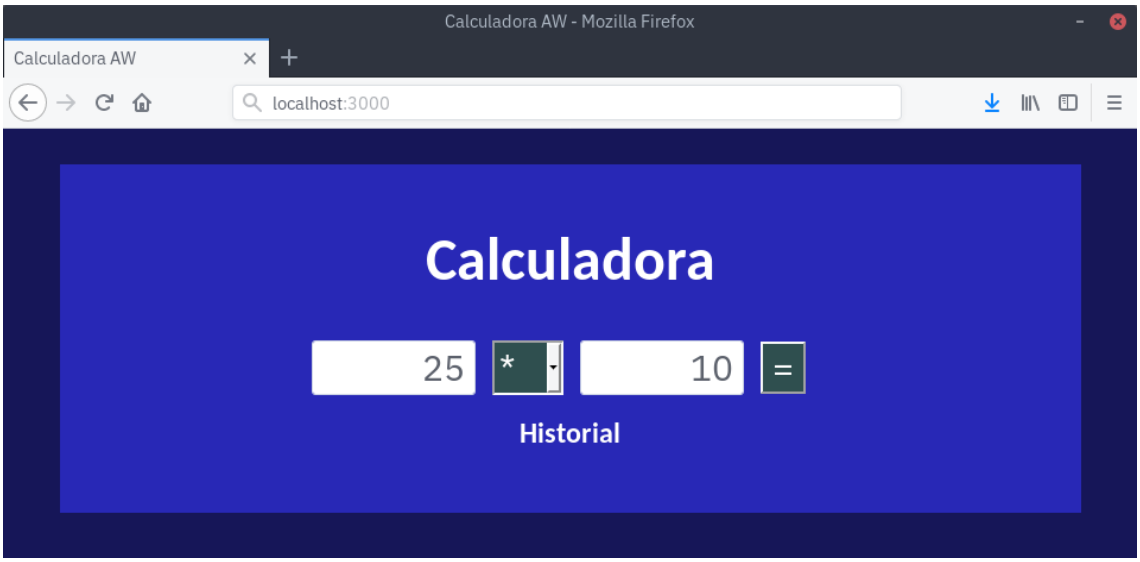


Figura 2: Calculadora web.

▷ 2. Aplicaciones web con *Express.js* y *EJS*

La página mostrada en la Figura 2 contiene un formulario con dos cuadros de texto, una lista desplegable entre ellos, y un botón `[=]`. El objetivo es realizar una aplicación web que aplique la operación seleccionada en la lista desplegable sobre los dos números indicados en los cuadros de texto. Las operaciones disponibles son cuatro: suma (+), resta (−), multiplicación (*) y división (/).

(a). *[2 pt]* Crea una aplicación web en el lado del servidor que reciba la información de este formulario y devuelva la misma página de la Figura 2 con el resultado de la operación a la derecha del botón `[=]`. Si alguno de los campos del formulario estuviese vacío, no fuese un número, o se intentase realizar una división entre cero, debe aparecer la palabra ERROR en lugar del resultado.

(b). *[2 pt]* Modifica la aplicación del apartado anterior para que almacene el historial de las operaciones realizadas por el usuario y lo muestre desde la operación más reciente hasta la más antigua (ver Figura 3). Utiliza, para ello, atributos de sesión. A la derecha de cada entrada del historial existe un botón `[Borrar]` que debe eliminarla. Las operaciones que hayan devuelto ERROR *no* deben aparecer en el historial.

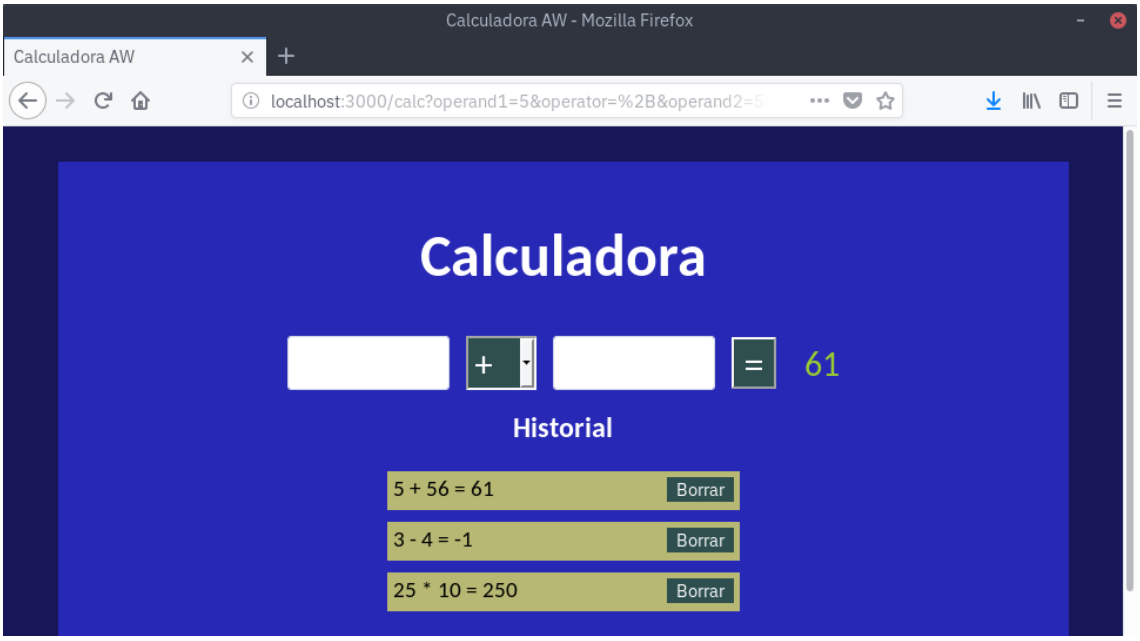


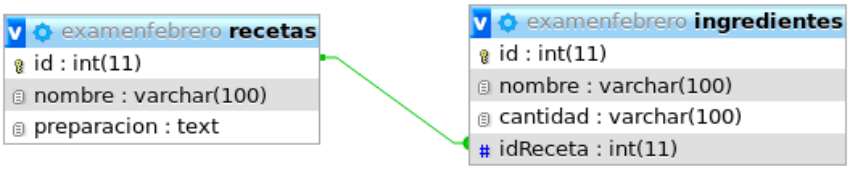
Figura 3: Calculadora web con historial.

Instrucciones para el ejercicio 2

- Añade los manejadores de ruta que sean necesarios en `ej2.js`. Puedes incorporar a la aplicación el *middleware* que creas necesario.
- El fichero estático `public/calculadora_template.html` puede ser de ayuda en la realización de la plantilla EJS.
- Para comprobar tu solución utiliza el *script* `ejercicio2.cmd`, que arranca el servidor utilizando nodemon.

▷ 3. Acceso a BD mediante *Node.js*

[2 pt] Supongamos el siguiente esquema relacional para una base de datos de recetas de cocina:



Cada receta tiene un nombre, unas instrucciones de preparación y una lista de ingredientes. Las columnas `id` tienen el atributo `AUTO_INCREMENT`.

En Javascript representamos las recetas mediante objetos de tres atributos: `nombre`, `preparacion` e `ingredientes`. El último de ellos es un array de objetos, cada uno con un atributo `nombre` y un atributo

Figura 4: Teclado para introducir contraseña.

`cantidad`. Implementa una función `insertarReceta(receta, callback)` que añada una receta a la BD. Una vez que los cambios se hayan realizado en la BD, debe llamarse a la función `callback`, pasando como primer parámetro un objeto `Error` si se ha producido algún error durante la inserción. Presta atención al problema de las $n + 1$ consultas.

Instrucciones para el ejercicio 3

- Realiza este ejercicio en el fichero `ej3.ejs`. Puedes ejecutar este mismo fichero para comprobar que tu solución es correcta. Puedes utilizar, para ello, el script `ejercicio3.cmd`.

▷ 4. AJAX y *jQuery*

Partiendo de un fichero HTML que muestra un teclado como el de la Figura 4:

(a). *[2 pt]* Utiliza *jQuery* para permitir que el usuario introduzca un código de cuatro dígitos mediante los botones numéricos. El recuadro de la parte superior debe mostrar la combinación introducida hasta el momento. Al introducir el cuarto dígito debe mostrarse un `alert()` con los cuatro dígitos introducidos. Tras esto, el usuario puede introducir otro número de cuatro dígitos.

(b). *[1 pt]* Extiende el ejemplo anterior para que tras mostrar el `alert()` realice una petición AJAX al siguiente servicio web:

Método: `GET`

URL: `/checkPassword`

Parámetros de entrada: Una variable `password` con el código de cuatro dígitos introducido por el usuario.

Códigos de respuesta: `200` (OK).

Resultado: Un objeto JSON con un atributo booleano `valid`. Este atributo tendrá el valor `true` si la contraseña es correcta, o `false` en caso contrario.

Tras realizar la petición, se debe llamar a la función `accesoPermitido()` o `accesoDenegado()` en función de la respuesta proporcionada por el servidor.

Instrucciones para el ejercicio 4

- El código HTML se encuentra en `public/ej4.html`, y su fichero Javascript correspondiente en `public/js/ej4_client.js`.
- El fichero `ej4.js` del directorio raíz implementa la funcionalidad del lado del servidor. Puedes arrancarlo mediante el *script* `ejercicio4.cmd`. Este servidor devuelve `{ valid : true }` cuando se le envía la contraseña `5665`.

Para entregar:

☐ Comprueba que has rellenado el fichero `Alumno.txt`.

☐ Crea un fichero comprimido (`.zip`) con el proyecto *Node* (incluyendo la carpeta `node_modules`)

☐ El nombre del fichero tiene que ser de la forma *DNI_Apellido*`.zip`. Incluye la letra del DNI y solamente el primer apellido.

☐ Entrega el fichero `.zip` utilizando el enlace del escritorio *Examenes en LABs - Entregas*. En la carpeta que aparece seleccionar *ALUMNOS entrega de prácticas y exámenes* y subir el `.zip`.

☐ **Antes de cerrar sesión**, dirígete al puesto del profesor para comprobar que el fichero se ha subido correctamente.