

### Ejercicio 0

Servlet que se reenvía a sí mismo, que consta de:

- Un formulario con una lista desplegable con nombres de libros y un botón de envío
  - Una lista con los nombres de libros que se van eligiendo
1. Los nombres de los libros del cuadro select corresponden a una propiedad estática de una clase java de nombre Catalogo (será un array de Strings inicializado en la propia declaración)
  2. El proceso está controlado por una variable de sesión (sólo una), que es un ArrayList de nombres de libros. Según enviemos el formulario, se actualizará dicha variable de sesión
  3. Si un libro ya forma parte del ArrayList de la sesión, no se volverá a agregar y se mostrará un mensaje de error: "Ya has elegido *NOMBRE\_DE\_LIBRO*"
  4. El cuadro desplegable debe retener el último libro seleccionado.

Guerra y paz ▼

AGREGAR

**No se han elegido libros**

Guerra y paz ▼

AGREGAR

**TU ELECCION:**

- Guerra y paz

Madame Bovary ▼

AGREGAR

**TU ELECCION:**

- Guerra y paz
- Madame Bovary

etc

- Agrega un mecanismo (botón, enlace, .....), para que el usuario pueda anular la sesión y empezar de 0

## Ejercicio 1

Aplicación web para adivinar una palabra secreta (elegida al azar de un almacén de palabras)

- La palabra oculta se toma al azar de una propiedad estática de un bean **AlmacenPalabras**
- En un principio aparecen todas sus letras ocultas
- El usuario puede desvelar algunas letras/posiciones de la palabra pinchando en el enlace VER. El Nº de vidas o Nº máximo de posiciones a desvelar es la mitad de la longitud de la palabra.
- En todo momento se mostrarán las vidas restantes
- Mediante un formulario se podrá comprobar la respuesta.
  - Si no se acierta y quedan vidas podremos continuar con el juego
  - Si se acierta o se agotan las vidas, finaliza el juego: se muestra el resultado de la jugada y se comienza otro juego con una nueva palabra
- **ServletJuego** es el servlet encargado de mostrar el juego, y el punto de partida del proyecto
- **ServletComprobar** se encarga de las comprobaciones (Procesa los enlaces y el formulario)

### ServletJuego

- Ejemplo: con palabra secreta de 11 letras: el servlet muestra una tabla con 11 enlaces, que se irán convirtiendo en texto (la letra correspondiente) a medida que se pinchen

Aspecto inicial

<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>
------------	------------	------------	------------	------------	------------	------------	------------	------------	------------	------------

- 5 vidas restantes

- Tu respuesta

Aspecto después de pinchar el Ver de la segunda y sexta letra

<u>Ver</u>	A	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	M	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>
------------	---	------------	------------	------------	---	------------	------------	------------	------------	------------

- 3 vidas restantes

- Tu respuesta

Aspecto después de pinchar  sin acertar la palabra y sin vidas restantes

Has perdido! La palabra era SALTAMONTES

<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>	<u>Ver</u>
------------	------------	------------	------------	------------	------------	------------	------------

- 4 vidas restantes

- Tu respuesta

Nuevo juego

**ServletComprobar:** No muestra nada. Deja preparados los atributos necesarios en el ámbito necesario (request o session) para volver al ServletJuego a mostrar los resultados

## Ejercicio 2

Clase **Pregunta**: Representa una pregunta tipo test

- Propiedades: el enunciado, una pista, sus posibles respuestas en un array, el nº de la respuesta correcta (posición o índice del array)
- Métodos: Constructor, y los que vayan surgiendo

Clase **Test**: Representa un test con varias preguntas tipo test

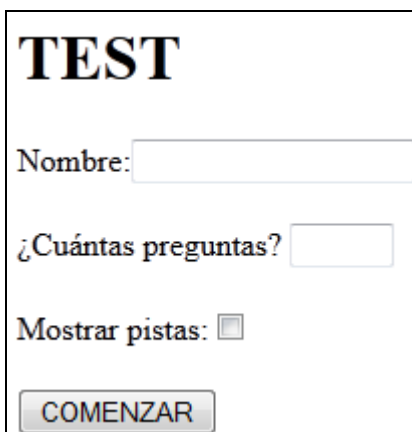
- Propiedades: nº de preguntas, ArrayList de Preguntas, array estático con varias Pregunta's preparadas
- Métodos:
  - constructor (int cant), que prepara un test con la cantidad de preguntas indicadas en el argumento (Las preguntas se seleccionan al azar de entre las existentes en el array estático, cuidando que no se repitan, y se pasan al arrayList). Si la cantidad recibida supera el Nº de preguntas disponible, se creará un test con todas las preguntas disponibles.
  - int comprobar(ArrayList de Integer), que recibe un ArrayList con los Nºs de respuesta de un usuario a las preguntas del test (en el mismo orden en que aparecen formuladas) y devuelve la cantidad de aciertos.

Otros que vayan surgiendo

La **parte web** consta de:

- Una página **index.html** de entrada con un formulario para que el usuario facilite 3 datos: su nombre, con cuántas preguntas instanciar el test, y si desea o no mostrar las pistas de cada pregunta
- Un servlet **ProcesoPregunta** accesible a través de la ruta virtual /servlets/ProcesoPregunta
- Un servlet **Resultado** accesible a través de la ruta virtual /servlets/Resultado, que mostrará el resultado del test (aciertos) y el tiempo utilizado

**index.jsp** es el acceso al test. Se trata de un simple formulario que se envía al servlet ProcesoPregunta



The image shows a web form titled "TEST" in a large, bold, serif font. Below the title, there are three input fields: "Nombre:" followed by a text box, "¿Cuántas preguntas?" followed by a text box, and "Mostrar pistas:" followed by a checkbox. At the bottom of the form is a button labeled "COMENZAR". The form is enclosed in a simple rectangular border.

El servlet **ProcesoPregunta** es el centro del proceso:

- Creará y mantendrá una sesión con los atributos que consideres necesarios. Ten en cuenta que todos aquellos datos a conservar entre envíos (excepto los campos de formulario) se guardarán en la sesión. Piensa bien qué datos debes conservar
- Mostrará cada vez una pregunta del test, con su pista, si así lo ha solicitado el usuario. En cada nuevo envío se guardará la respuesta de la anterior y se mostrará la siguiente
- En la última pregunta del test, el botón de submit mostrará "FIN"
- La última pregunta no debe ser enviada al mismo servlet, sino al servlet **Resultado**

Se irá reenviando a sí mismo, mostrando cada vez una nueva pregunta.

(Cuando para una pregunta no se seleccione ningún radio, se volverá a formular la misma pregunta, mostrando un mensaje de error en la parte superior)

<b>DIRECTOR DE 'Psicosis':</b>  <input type="radio"/> Alfred Hitchcock <input type="radio"/> Stanley Kubrick <input type="radio"/> Orson Welles  <input type="button" value="SIGUIENTE"/>  <i>* PISTA: Era británico</i>	<b>¿PELICULA MAS OSCARIZADA?</b>  <input type="radio"/> Amadeus <input type="radio"/> Benhur <input type="radio"/> West Side Story <input type="radio"/> El Padrino II  <input type="button" value="SIGUIENTE"/>  <i>* PISTA: 11 oscar en 1959</i>	.....	<b>EL CINEMATOGRAFO FU</b>  <input type="radio"/> Thomas Alva Edison <input type="radio"/> Hermanos Lumiere  <input type="button" value="FIN"/>  <i>* PISTA: Era británico</i>
--	---	-------	---

El servlet **Resultado** mostrará la cantidad de aciertos, un mensaje de retroalimentación acorde a dicha cantidad, el tiempo invertido en completar el test y un enlace a la página index.jsp

<b>Pedro, has acertado 8 preguntas de un total de 10</b>  Tienes muy buenos conocimientos de cine  Tiempo de respuesta: 4 minutos 15 segundos  <a href="#">NUEVO INTENTO</a>
--

### Ejercicio 3

index.jsp

## INICIA TU APUESTA

Nombre:

index.jsp se dirige al servlet **EscribeApuesta**

### Juan González, escribe tu apuesta:

Valladolid	Espanyol	<input type="button" value="▼"/>
Atletico de Madrid	Málaga	<input type="button" value="▼"/>
Betis	R.Sociedad	<input type="button" value="▼"/>
Ath.Club	Osasuna	<input type="button" value="▼"/>
Levante	Valencia	<input type="button" value="▼"/>
Zaragoza	Getafe	<input type="button" value="▼"/>

Este servlet muestra una tabla con la típica quiniela de fútbol:

- Los pares de equipos se extraen de un fichero de texto formado por líneas con pares de equipos
  - El nombre de este fichero es un parámetro que se pasa en la configuración del servlet
  - Haz lo necesario para que el fichero sea leído 1 sola vez durante la vida del servlet (evitar que cada petición del servlet tenga que releer el fichero)
- Los combos tienen las opciones 1, X, 2 y una opción vacía

**EscribeApuesta** se dirige al servlet **ProcesaApuesta**

- ProcesaApuesta** crea una variable de sesión (un array de Strings )con la apuesta elegida
  - Si la apuesta no es completa (algún campo en blanco) vuelve a EscribeApuesta mediante forward para completarla
  - Si es completa, la muestra e incluye un enlace al servlet anterior para poder modificarlaEn cualquier caso, debes modificar el servlet EscribeApuesta para que conserve los valores elegidos, basándose en los datos de la sesión. E incorpore un mensaje de error al lado de los vacíos si los hubiera

Juan González, escribe tu apuesta:

Valladolid	Espanyol	<input type="button" value="2 ▼"/>
Atletico de Madrid	Málaga	<input type="button" value="1 ▼"/>
Betis	R.Sociedad	<input type="button" value="X ▼"/>
Ath.Club	Osasuna	<input type="button" value="2 ▼"/>
Levante	Valencia	<input type="button" value="X ▼"/>
Zaragoza	Getafe	<input type="button" value="1 ▼"/>

## Apuesta guardada

Valladolid:Espanyol 2  
Atletico de Madrid:Malaga 1  
Betis:R.Sociedad X  
Ath.Club:Osasuna 2  
Levante:Valencia X  
Zaragoza:Getafe 1  
[REVISAR LA APUESTA](#)

