

Reglas de la prueba

- No se puede modificar el HTML ni el CSS a mano.
- No se pueden dejar etiquetas vacías. Si añades una etiqueta al DOM y ya no es necesaria, será eliminada del mismo.
- Está prohibido usar `innerHTML` e insertar elementos vacíos (un párrafo sin contenido, por ejemplo). Sí se puede usar `textContent`.

Nota: si alguna parte del examen no se consigue realizar, se puede pasar a la siguiente parte. Por ejemplo: si no se sabe realizar la validación se puede pasar a la siguiente parte dando por hecho que los datos son correctos. Si no se sabe cargar dinámicamente el contenido del fichero se puede crear a mano una variable con el contenido para poder continuar el examen. Esto implica un 0 en la parte que se ha saltado, pero permite continuar el examen. **Indícalo claramente en un comentario** al principio del fichero de código.

Contexto

En una empresa nos encargan completar la aplicación web para llevar el control de los portes de una jornada que realiza un autónomo con un camión dedicado al transporte de alimentos.

Las mercancías se llevan en dos **tipos de contenedores: palés o boxes**.

- Cada box se cobra a 25€ el porte. Cada palé se cobra a 175€ el porte.

Las mercancías dentro de los contenedores pueden ser de tres tipos: frutas, verduras y carne. La única regla es que **si hay carne en un contenedor de un porte, no puede haber otro contenedor con verdura o fruta** para evitar contaminaciones cruzadas.

Parte 1: Carga de datos desde un archivo

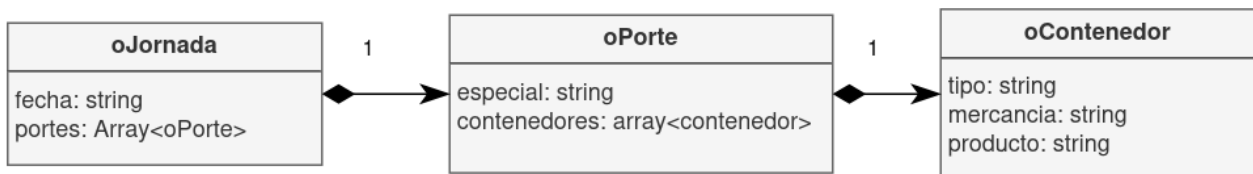
El camión de reparto ya ha realizado algunos portes que se han almacenado y pueden ser recuperados a través de una API que nos devolverá los portes registrados en formato XML. Realiza una petición de tipo GET al cargar la página para obtener estos datos iniciales.

Fragmento del fichero `portes.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<jornada fecha="07/03/2025">
  <porte especial="especial">
    <palé mercancia="fruta">Melón</palé>
    <palé mercancia="fruta">Sandía</palé>
    <box mercancia="verdura">Calabacín</box>
  </porte>
  ...
</jornada>
```

Cuando cargues el código de tu aplicación crea un objeto `oJornada` con los datos del fichero que te permita usarlo durante el resto del ejercicio sin tener que leer de nuevo el fichero. Contendrá dos campos:

- La **fecha** de la jornada.
- Un **array de objetos** en el cada objeto representa un **porte** dado en esa fecha.
 - Un porte consta de un indicador de si es **especial** y un **array de contenedores**.
 - Un **contenedor** se representa con un objeto con tres opciones:
 - **tipo**: *palé* o *box*
 - **mercancia**: *fruta*, *verdura* o *carne*
 - **producto**: el producto concreto, por ejemplo *Plátanos*



Nota: Si se produce un error leyendo el fichero, simplemente muéstralo en la consola con un mensaje de error. La carga del fichero se hará mediante una función asíncrona como si estuviéramos consumiendo una API de otro servidor.

Nota: Si no sabes realizar este apartado, lee el fichero `portes.estranguificados` donde tendrás los datos para poder continuar el resto del examen aunque no hayas hecho esta parte.

Parte 2: Modificación del CSS y carga dinámica de código

Cuando se pulse en el botón *Cambiar tema* se mostrará en un `<dialog>` un formulario con un párrafo y dos botones:

- El párrafo preguntará *¿Desea cambiar el tema de la web?*
- Botón *Cerrar* (que cerrará el cuadro de diálogo y lo eliminará del DOM)
- Botón *Cambiar tema* que cambiará el tema cada vez que se pulsa como se indica más abajo abajo y mostrará un *toast* indicando que el tema se ha cambiado.

Cuando se pulse por primera vez el botón de *Cambiar tema* del cuadro de diálogo se cargará dinámicamente un fichero javascript llamado `gestionarCSSyToast.js` que contendrá dos funciones.

La primera función se llamará `cambiarColorAcento()` y hará lo siguiente:

- Cuando se ejecute calculará aleatoriamente un color. **Nota:** puedes usar `rgb(red, green, blue)` para codificar el color o hexadecimal.
- Modificará la regla CSS existente del fichero CSS para cambiar el color de acento de la web (variable `--accent-color`).
- **Nota:** Si no sabes modificar la regla, elimínala y vuelvela a crear en la misma posición.

La segunda función será `showToast(mensaje)` recibirá como parámetro una cadena de caracteres (un `mensaje`) y hará lo siguiente:

- Añadirá al principio de la `<div id="divToast">` una división con `width: 250px`, `padding: 1rem` y color de fondo celeste.
- El contenido de la división será un párrafo con el mensaje.
- Se establecerá un temporizador que hará que se elimine la división en 3 segundos automáticamente.

Ese fichero `gestionarCSSyToast.js` se cargará una única vez aunque se repita la operación.

Parte 3: Validación

Botón Inicio

Al pulsar el botón *Inicio*, si el contenido de la `<div id="landingPage">` se encuentra oculto tendrá que mostrarse y ocultarse la división `<div id="crearPorte">`.

Botón Portes

Cuando se pulse el botón de *Portes* el contenido de la `<div id="landingPage">` tendrá que ocultarse y mostrarse a división `<div id="crearPorte">`.

Cada porte está compuesto por una serie de contenedores (palés o boxes) con mercancía. Crea un objeto de tipo `oPorte` que contendrá los datos de los contenedores del porte (para poder añadirlo al objeto `oJornada` posteriormente). Este objeto irá llenándose con los datos de cada box o palé que se le añada.

En la interfaz que se muestra hay dos botones:

- Añadir contenedores al porte (al objeto `oPorte`).
- Añadir el porte (el objeto `oPorte`) con los contenedores indicados al objeto `oJornada`.

Cuando se pulse el botón *Añadir contenedores al porte* tendrá que validarse lo siguiente:

- El *tipo de contenedor* (box o palé) se seleccionará mediante los *radiobutton*. Haz que por defecto esté seleccionado el box.
- El *tipo de mercancía* (fruta, verdura o carne) se seleccionará mediante otros *radiobutton*. Por defecto estará seleccionada la fruta.
- Todos los campos tienen que estar rellenos.
- El *producto* debe ser una cadena de 20 caracteres como máximo. El primer carácter tendrá que ser una letra mayúsculas, seguidas de letras minúsculas (incluyendo vocales acentuadas, con diéresis, eñes y cedillas, pero ningún otro carácter) y puede terminar con

números (sólo pueden aparecer al final). No se permitirán caracteres de puntuación (paréntesis, exclamaciones, comas, puntos, comillas...) ni símbolos (como el ampersand, el signo del euro, almohadilla, porcentajes, guiones bajos, espacios...).

- El checkbox *Porte especial* puede estar marcado o no.
- El número de contenedores ha de ser positivo e indicará el número de boxes o palés con esa mercancía que se desean añadir.
- Si en el porte ya se ha añadido un contenedor con mercancía de `carne`, no se podrá añadir una de `fruta` o de `verdura` o viceversa.

Si hay algún tipo de error se mostrarán en un `<dialog>`, recuerda que los mensajes de error deben ser descriptivos y con el tono adecuado.

Si la validación es correcta, se añadirán los contenedores al objeto `oPorte` y mostraremos un mensaje al usuario usando un `showToast(mensaje)` para indicar que se añadieron los contenedores al porte.

Cuando se pulse el botón de *Añadir porte a la jornada* se añadirá el nuevo porte contenido en `oPorte` al objeto `oJornada`. Si el objeto `oPorte` no contiene ningún contenedor no se hará nada.

Opcional: se realizará una petición de tipo POST a `/api/portes` mandando el objeto `oJornada`. Evidentemente esto fallará porque no tenemos implementado el servidor con una API REST, de modo que mostraremos un mensaje al usuario usando un `showToast(mensaje)` para indicar el fallo del servidor, pero lo dejaremos preparado para que indique que se ha añadido el nuevo porte correctamente si no hubo error.

Parte 4: Manipulación del DOM y evento delegado

Cuando se lea el fichero `portes.xml` y cada vez que se añada un nuevo porte, se creará una lista dentro de la división `<div id="crearPorte">` con los portes contenidos en el objeto `oJornada`, añadiendo la posición que ocupa en el array en un atributo `data-x`. Algo así:

- Porte 1
- Porte 2
- Porte 3

Cuando se seleccione un elemento de la lista, con un evento delegado se creará una tabla debajo con los datos del porte seleccionado.

Suponiendo que se ha seleccionado el Porte 2:

+-----+-----+-----+-----+			
Porte	2	Fecha	10/03/2025
+-----+-----+-----+-----+			
Producto		Tipo	Precio

+-----+	+-----+	+-----+	+-----+
Zanahoria	Palé	175 €	
+-----+	+-----+	+-----+	+-----+
Sandía	Palé	175 €	
+-----+	+-----+	+-----+	+-----+
Melocotón	Box	25 €	
+-----+	+-----+	+-----+	+-----+
Kiwi	Box	25 €	
+-----+	+-----+	+-----+	+-----+
Lechuga	Box	25 €	
+-----+	+-----+	+-----+	+-----+
Lechuga	Box	25 €	
+-----+	+-----+	+-----+	+-----+
Especial: no	Total	450 €	
+-----+	+-----+	+-----+	+-----+

El total se calcula sumando el precio de cada palé y cada box.
Cuando se añada un nuevo porte a la aplicación, se actualizará la lista y se borrará la tabla.

Puntuación

- Parte 1) Carga de datos: 2 puntos
- Parte 2) CSS y carga dinámica de código: 2,5 puntos
- Parte 3) Validación: 3 puntos + 0.5 opcional
- Parte 4) Manipulación del DOM y evento delegado: 2,5 puntos