

# Examen de JavaScript

## Índice

Examen de JavaScript.....	1
Contexto común.....	1
E) DOM + eventos + formularios + validación + almacenamiento (25 min).....	1
F) fetch + render dinámico + manejo de errores (20 min).....	2
Rúbrica.....	3

## Contexto común

- Trabaja en script.js con "use strict".
- Documenta con comentarios breves cada bloque.
- Valida en consola sin errores.
- No uses librerías externas.

## E) DOM + eventos + formularios + validación + almacenamiento (25 min)

**HTML base asumido. Adjunto al examen.**

- Form con #frmAlumno
- campos del formulario : #nombre, #email, #edad.
- Lista destino #listaAlumnos.
- Botones #btnGuardar y #btnLimpiar.

### Enunciado

1. Captura submit del formulario. Prevén envío por defecto.
2. Valida:
  - nombre: mínimo 2 letras, solo letras y espacios.
  - email: expresión regular, estándar sencillo.
  - edad: entero 16–99.Muestra errores bajo cada campo con `<small class="error">...</small>` y añade/remueve una clase `is-invalid`.
3. Si es válido, crea `<li>` con plantilla: "Nombre – email – edad", añádelo a `#listaAlumnos`.
4. Guarda en `localStorage` la lista completa como JSON. Al cargar la página, restaura la lista.
5. `#btnLimpiar` borra la lista y limpia `localStorage`.
6. Cambia el aspecto: cuando el foco entra en un input, añade clase `focus`; al salir, la quita.

**Pistas:**      `addEventListener`,      `classList.add/remove`,      `insertAdjacentHTML`      o  
`createElement+appendChild`, `localStorage.getItem/setItem`.

**RA/CE:**      RA2: h      RA3: d, e, g, h      RA5: a, b, c, d, e, f, g, h      RA6: a, b, c, d, e, h

## F) fetch + render dinámico + manejo de errores (20 min)

### Datos

Los servirá el profesor en la url: **http://10.103.255.0:3000/productos**

Los datos son un array de objetos con la estructura: [{ "id":1, "title":"Teclado", "price":19.99 }, ...]

### Enunciado

1. Crea async function cargarProductos(url) que haga:
  - fetch(url)
  - Comprobación resp.ok y throw con mensaje emergente alert si falla
  - await resp.json() y return.
  - try/catch que muestre mensaje de error en #estado.
2. Llama a cargarProductos(URL) al cargar la página.
3. Renderiza en #productos una lista <ul> con title – price usando toLocaleString("es-ES", {style:"currency",currency:"EUR"}).
4. Añade un buscador #txtFiltro. En input, filtra por title y re-renderiza sin hacer otro fetch.
5. Añade botón "Abrir detalles" por producto que abra window.open una ventana secundaria simple con el detalle del producto (id, título y precio) generado vía win.document; usa DOM en la ventana hija (win.document.body.append(...)). Controla si el popup está bloqueado: si win==null, muestra aviso.
6. Actualiza document.title con el número de productos y añade una clase al <body> cuando haya error de red

**RA/CE:** RA2: a, f, h RA3: a, b, c, d, e, f, h RA6: a, b, c, d, e, h RA7: a,b,c,e,f.

# Rúbrica

Criterio (RA)	Excelente (10)	Bien (7)	Suficiente (5)	Insuficiente (0)
<b>E1 — RA2-h (0,292 pts)</b> Prueba/entorno en E).	Sin errores; flujos verificados. (0,292)	Algún aviso. (0,204)	Verificación mínima. (0,146)	Sin verificación. (0,000)
<b>E2 — RA3-d (0,292 pts)</b> Generar HTML desde código.	Crea <li> con plantilla correcta y añade a #listaAlumnos. (0,292)	Inserta sin plantilla o con formato menor. (0,204)	Inserta texto plano. (0,146)	No inserta. (0,000)
<b>E3 — RA3-e (0,292 pts)</b> Interacción con usuario.	Mensajes de error bajo cada campo con clase is-invalid. (0,292)	Mensajes parciales. (0,204)	Mensajes poco visibles. (0,146)	Sin mensajes. (0,000)
<b>E4 — RA3-g (0,292 pts)</b> Almacenamiento navegador.	Guarda y restaura lista en localStorage. (0,292)	Guarda o restaura, no ambos. (0,204)	Persistencia inestable. (0,146)	No usa almacenamiento. (0,000)
<b>E5 — RA3-h (0,292 pts)</b> Depurar/documentar E).	Depura y documenta. (0,292)	Uno de los dos. (0,204)	Escaso. (0,146)	Nulo. (0,000)
<b>E6 — RA5-a (0,292 pts)</b> Captura de eventos.	Captura submit y previene envío. (0,292)	Captura con fallo menor. (0,204)	Captura parcial. (0,146)	No captura. (0,000)
<b>E7 — RA5-b (0,292 pts)</b> Gestión de eventos.	Maneja focus/blur y botones según enunciado. (0,292)	Falta un manejador. (0,204)	Manejo básico. (0,146)	No maneja. (0,000)
<b>E8 — RA5-c (0,292 pts)</b> Tipos de eventos.	Usa submit, input, click, focus/blur con criterio. (0,292)	Falta un tipo. (0,204)	Tipos mínimos. (0,146)	Tipos incorrectos. (0,000)
<b>E9 — RA5-d (0,292 pts)</b> Código que captura y usa eventos.	Respuesta de UI coherente en todos los flujos. (0,292)	Un flujo incompleto. (0,204)	Respuesta parcial. (0,146)	No responde. (0,000)
<b>E10 — RA5-e (0,292 pts)</b> Gestión de formularios.	Valida nombre, email y edad con reglas fijadas. (0,292)	Falta una regla o borde. (0,204)	Validación superficial. (0,146)	Sin validación. (0,000)
<b>E11 — RA5-f (0,292 pts)</b> Validación con eventos.	Valida en tiempo real o al enviar con feedback claro. (0,292)	Solo al enviar o feedback pobre. (0,204)	Validación tardía. (0,146)	Sin validación vía eventos. (0,000)
<b>E12 — RA5-g (0,292 pts)</b> Expresiones regulares.	Email con regex estándar y controles. (0,292)	Regex básica sin anclas/opciones. (0,204)	Regex muy laxa. (0,146)	Sin regex. (0,000)
<b>E13 — RA5-h (0,292 pts)</b> Prueba/documentación E).	Casos probados y anotados. (0,292)	Parcial. (0,204)	Mínimo. (0,146)	Nulo. (0,000)
<b>E14 — RA6-a (0,292 pts)</b> Reconoce DOM.	Accede a nodos por selectores id/clase correctamente. (0,292)	Selección mejorable. (0,204)	Accesos imprecisos. (0,146)	No accede. (0,000)
<b>E15 — RA6-b (0,292 pts)</b> Objetos del modelo.	Usa classList, createElement, insertAdjacentHTML. (0,292)	Omite uno. (0,204)	Uso básico. (0,146)	Uso erróneo. (0,000)
<b>E16 — RA6-c (0,292 pts)</b> Acceso a documento.	Lee y valida valores de inputs sin errores. (0,292)	Lecturas parciales. (0,204)	Lecturas con conversiones dudosas. (0,146)	No accede. (0,000)

<b>Criterio (RA)</b>	<b>Excelente (10)</b>	<b>Bien (7)</b>	<b>Suficiente (5)</b>	<b>Insuficiente (0)</b>
<b>E17 — RA6-d (0,292 pts)</b> Crear/modificar estructura.	Crea <code>&lt;li&gt;</code> y limpia lista con botón. <b>(0,292)</b>	Crea o limpia, no ambos. <b>(0,204)</b>	Manipulación parcial. <b>(0,146)</b>	No manipula. <b>(0,000)</b>
<b>E18 — RA6-e (0,292 pts)</b> Asociar acciones a eventos DOM.	Listeners bien registrados y removidos si procede. <b>(0,292)</b>	Registro correcto sin limpieza. <b>(0,204)</b>	Registro disperso. <b>(0,146)</b>	No asocia. <b>(0,000)</b>
<b>E19 — RA6-h (0,292 pts)</b> Independencia de capas.	HTML semántico, CSS por clases, JS solo comportamiento. <b>(0,292)</b>	Algún acoplamiento leve. <b>(0,204)</b>	Mezcla puntual. <b>(0,146)</b>	Mezcla fuerte. <b>(0,000)</b>
<b>F1 — RA2-a (0,218 pts)</b> Selección del lenguaje cliente.	Implementa solución en JS cliente adecuada al problema. <b>(0,218)</b>	Adecuada con pequeñas carencias. <b>(0,153)</b>	Aceptable pero pobre. <b>(0,109)</b>	Inadecuada. <b>(0,000)</b>
<b>F2 — RA2-f (0,212 pts)</b> Herramientas de prueba.	try/catch, trazas claras y verificación en <code>#estado</code> . <b>(0,212)</b>	Falta una de las tres. <b>(0,148)</b>	Solo trazas. <b>(0,106)</b>	Sin control. <b>(0,000)</b>
<b>F3 — RA2-h (0,212 pts)</b> Prueba/entorno en F).	Sin errores; estados de red comprobados. <b>(0,212)</b>	Avisos menores. <b>(0,148)</b>	Pocas pruebas. <b>(0,106)</b>	Nulo. <b>(0,000)</b>
<b>F4 — RA3-a (0,212 pts)</b> Objetos predefinidos.	Usa window, document, location de forma correcta. <b>(0,212)</b>	Uso parcial. <b>(0,148)</b>	Uso básico. <b>(0,106)</b>	Mal uso. <b>(0,000)</b>
<b>F5 — RA3-b (0,212 pts)</b> Ventanas/navegador.	Abre popup con window.open, controla bloqueo. <b>(0,212)</b>	Sin control de bloqueo. <b>(0,148)</b>	Abre sin validar. <b>(0,106)</b>	No abre. <b>(0,000)</b>
<b>F6 — RA3-c (0,212 pts)</b> Cambios en navegador/documento.	Actualiza document.title según el nº de productos y aplica clase de error al <code>&lt;body&gt;</code> . <b>(0,212)</b>	Actualiza uno de los dos. <b>(0,148)</b>	Cambio parcial. <b>(0,106)</b>	No cambia. <b>(0,000)</b>
<b>F7 — RA3-d (0,212 pts)</b> Generación HTML.	Renderiza <code>&lt;ul&gt;</code> con title – price correctamente. <b>(0,212)</b>	Formato menor incorrecto. <b>(0,148)</b>	Texto plano. <b>(0,106)</b>	No renderiza. <b>(0,000)</b>
<b>F8 — RA3-e (0,212 pts)</b> Interacción con usuario.	Mensajes claros en errores y al bloquear popup. <b>(0,212)</b>	Mensajes escuetos. <b>(0,148)</b>	Mensajes confusos. <b>(0,106)</b>	Sin mensajes. <b>(0,000)</b>
<b>F9 — RA3-f (0,212 pts)</b> Varias ventanas.	Pinta detalles en la ventana hija usando DOM, sin document.write. <b>(0,212)</b>	Pinta pero con write o mezcla. <b>(0,148)</b>	Pinta incompleto. <b>(0,106)</b>	No pinta. <b>(0,000)</b>
<b>F10 — RA3-h (0,212 pts)</b> Depuración/documentación F).	Depura y documenta. <b>(0,212)</b>	Uno de los dos. <b>(0,148)</b>	Mínimo. <b>(0,106)</b>	Nulo. <b>(0,000)</b>
<b>F11 — RA6-a (0,212 pts)</b> Reconoce DOM en F).	Selecciones y contenedores correctos. <b>(0,212)</b>	Selección mejorable. <b>(0,148)</b>	Básico. <b>(0,106)</b>	Incorrecto. <b>(0,000)</b>
<b>F12 — RA6-b (0,212 pts)</b> Objetos del modelo.	Usa append, textContent, classList. <b>(0,212)</b>	Falta uno. <b>(0,148)</b>	Uso básico. <b>(0,106)</b>	Mal uso. <b>(0,000)</b>
<b>F13 — RA6-c (0,212 pts)</b> Acceso a documento.	Lee filtro <code>#txtFiltro</code> y dispara render. <b>(0,212)</b>	Lectura o evento con fallos menores. <b>(0,148)</b>	Funciona a veces. <b>(0,106)</b>	No funciona. <b>(0,000)</b>
<b>F14 — RA6-d (0,212 pts)</b> Crear/modificar estructura.	Re-render sin duplicar ni mutar originales. <b>(0,212)</b>	Algún duplicado. <b>(0,148)</b>	Re-render parcial. <b>(0,106)</b>	No re-renderiza. <b>(0,000)</b>
<b>F15 — RA6-e (0,212 pts)</b>	input para filtrar y otros	Falta un evento.	Asociación pobre.	Sin eventos. <b>(0,000)</b>

<b>Criterio (RA)</b>	<b>Excelente (10)</b>	<b>Bien (7)</b>	<b>Suficiente (5)</b>	<b>Insuficiente (0)</b>
<b>pts</b> ) Eventos del modelo.	eventos bien asociados. <b>(0,212)</b>	<b>(0,148)</b>	<b>(0,106)</b>	
<b>F16 — RA6-h (0,212 pts)</b> Independencia de capas.	Mantiene separación HTML/CSS/JS. <b>(0,212)</b>	Algún acoplamiento. <b>(0,148)</b>	Mezcla puntual. <b>(0,106)</b>	Mezcla fuerte. <b>(0,000)</b>
<b>F17 — RA7-a (0,212 pts)</b> Ventajas/inconvenientes async.	Manejo de estados y mensajes que reflejan pros/contras de async. <b>(0,212)</b>	Parcial. <b>(0,148)</b>	Superficial. <b>(0,106)</b>	Nulo. <b>(0,000)</b>
<b>F18 — RA7-b (0,212 pts)</b> Mecanismos disponibles.	Usa fetch con resp.ok y errores bien gestionados. <b>(0,212)</b>	Falta una comprobación. <b>(0,148)</b>	Manejo superficial. <b>(0,106)</b>	Sin control. <b>(0,000)</b>
<b>F19 — RA7-c (0,212 pts)</b> Objetos relacionados.	Emplea Request/Response y JSON correctamente. <b>(0,212)</b>	Uso parcial. <b>(0,148)</b>	Uso básico. <b>(0,106)</b>	Mal uso. <b>(0,000)</b>
<b>F20 — RA7-e (0,212 pts)</b> Actualización dinámica.	Render dinámico desde datos remotos y filtro sin nuevo fetch. <b>(0,212)</b>	Render correcto pero refetch innecesario. <b>(0,148)</b>	Render parcial. <b>(0,106)</b>	No actualiza. <b>(0,000)</b>
<b>F21 — RA7-f (0,206 pts)</b> Formatos de intercambio.	Trata JSON sin errores de parseo y con validación. <b>(0,206)</b>	Parseo sin validación. <b>(0,144)</b>	Parseo frágil. <b>(0,103)</b>	Falla el parseo. <b>(0,000)</b>