# ÍNDICE

[ÍNDICE 1](#_Toc210722383)

[TEMA // RA-CE 2](#_Toc210722384)

[**ACTIVIDAD (2a, 2b, 2c, 2d, 2e, 2f, 2g, 2h)** 3](#_Toc210722385)

[Objetivo 3](#_Toc210722386)

[Enunciado 3](#_Toc210722387)

[Proyectos Propuestos 4](#_Toc210722388)

[Requisitos de Entrega 6](#_Toc210722389)

[Checklist de entrega (para el alumnado) 7](#_Toc210722390)

# TEMA // RA-CE

|  |  |  |
| --- | --- | --- |
| **UD** | **RA** | **CE** |
| 2. Introducción a la programación en JavaScript | RA2. Escribe sentencias simples, aplicando la sintaxis del lenguaje y verificando su ejecución sobre navegadores web. | 2.a) Se ha seleccionado un lenguaje de programación de clientes web en función de sus posibilidades. |
| 2.b) Se han utilizado los distintos tipos de variables y operadores disponibles en el lenguaje. |
| 2.c) Se han identificado los ámbitos de utilización de las variables. |
| 2.d) Se han reconocido y comprobado las peculiaridades del lenguaje respecto a las conversiones entre distintos tipos de datos. |
| 2.e) Se han utilizado mecanismos de decisión en la creación de bloques de sentencias. |
| 2.f) Se han utilizado bucles y se ha verificado su funcionamiento. |
| 2.g) Se han añadido comentarios al código. |
| 2.h) Se han utilizado herramientas y entornos para facilitar la programación, prueba y documentación del código. |

# **ACTIVIDAD (2a, 2b, 2c, 2d, 2e, 2f, 2g, 2h)**

## Objetivo

El objetivo de esta actividad es **aplicar de forma práctica los fundamentos de la programación en JavaScript** en un proyecto sencillo que integre los conocimientos básicos de la unidad.

Concretamente, se pretende:

* Seleccionar y utilizar JavaScript como lenguaje de programación en cliente (2.a).
* Implementar correctamente **variables y operadores** en distintos contextos (2.b).
* Reconocer y aplicar los **ámbitos de las variables** (var, let, const) (2.c).
* Identificar y comprobar las **conversiones de tipos de datos** (2.d).
* Usar **estructuras de decisión** (if, switch) para controlar el flujo de ejecución (2.e).
* Usar **bucles** (while, do while, for) para repetir instrucciones y comprobar su funcionamiento (2.f).
* Incluir **comentarios claros y adecuados** que documenten el código (2.g).
* Utilizar **herramientas y entornos** de programación (editor, navegador, consola, control de versiones) para probar, documentar y presentar el trabajo (2.h).
* **Aplicar principios básicos de diseño web**, cuidando la selección de colores y tipografías para garantizar la legibilidad y la coherencia visual del proyecto.

En conjunto, esta actividad permitirá al alumnado **construir un proyecto completo en JavaScript**, documentarlo y presentarlo, reforzando tanto las competencias técnicas como las comunicativas, además de fomentar la sensibilidad hacia aspectos de diseño que enriquecen el resultado final.

## Enunciado

El alumnado, organizado en grupos de **2–3 personas**, deberá desarrollar un **proyecto en JavaScript** que integre los contenidos de la unidad.

* El proyecto se elegirá de entre una **lista de propuestas** facilitadas por el profesor.
* Los **grupos quedarán registrados** por el profesor, procurando que sean los mismos que para el proyecto integrado.
* El trabajo deberá cubrir tanto los **aspectos técnicos de programación** (uso de variables, operadores, estructuras de control, bucles, comentarios, etc.) como un **mínimo de criterios de diseño visual** (colores y tipografías coherentes con el proyecto).
* El resultado será **presentado en clase**, de forma organizada y siguiendo una estructura similar a la del **proyecto integrado final**, incluyendo:
  + **Funcionalidad final** de la aplicación.
  + **Estructura y organización** del proyecto.
  + **Gestión y organización del trabajo en equipo** durante el desarrollo.
  + **Funcionamiento del código** y explicación de los aspectos más relevantes, destacando las dificultades encontradas y cómo se resolvieron.
  + **Decisiones de diseño** (tipografía, colores, etc.) y **herramientas utilizadas** durante el desarrollo.
  + **Uso de GitHub y Jira (si procede):** se valorará positivamente que el equipo muestre cómo ha organizado el **control de versiones en GitHub** y la **planificación de tareas en Jira**, siempre que estas herramientas se hayan empleado en el proyecto.

### Proyectos Propuestos

#### 1. Gestor de notas escolares

**Requisitos funcionales:**

1. Permitir **introducir varias asignaturas con sus calificaciones**.
2. Calcular automáticamente la **media** de todas las notas introducidas.
3. Mostrar si el alumno **aprueba o suspende** en función de la media.
4. Clasificar el resultado final en **Sobresaliente, Notable, Aprobado o Suspenso** mediante estructuras de decisión (if o switch).
5. Presentar los resultados en una **tabla coloreada**, aplicando un color distinto a cada rango de nota para facilitar la interpretación.
6. **Aplicar principios de diseño web**: seleccionar colores adecuados para la legibilidad y tipografías coherentes para la presentación de resultados.

#### 2. Juego de adivinanza de números

**Requisitos funcionales:**

1. Generar un **número aleatorio entre 1 y 100** al iniciar el juego.
2. Permitir que el usuario introduzca intentos para **adivinar el número**.
3. Mostrar **pistas**: “El número es mayor” o “El número es menor”, utilizando condicionales.
4. Llevar un **contador de intentos** e incrementar su valor en cada jugada.
5. Mostrar un **mensaje final de valoración** en función del número de intentos realizados (por ejemplo, Excelente, Bien, Regular).
6. **Aplicar principios de diseño web**: la interfaz deberá mostrar cambios de colores según el progreso (verde al acertar, rojo al fallar) y utilizar tipografías adecuadas para una buena experiencia visual.

#### 3. Agenda de contactos

**Requisitos funcionales:**

1. Permitir **añadir nuevos contactos** con nombre, teléfono y correo electrónico.
2. Evitar **duplicados**, comprobando que no se registre dos veces un contacto con el mismo nombre.
3. Incluir una opción de **búsqueda** de contactos por nombre.
4. Mostrar la **lista completa de contactos ordenada alfabéticamente**, usando bucles para recorrer la información.
5. Permitir **eliminar contactos** de la agenda.
6. **Aplicar principios de diseño web**: presentar el listado con colores diferenciados para cabeceras y secciones, y tipografía clara que facilite la lectura.

#### 4. Gestor de tareas pendientes (To-Do List)

**Requisitos funcionales:**

1. Permitir **añadir nuevas tareas** a una lista.
2. Mostrar en pantalla todas las **tareas registradas**.
3. Permitir **eliminar tareas** de la lista.
4. Posibilitar marcar tareas como **“hechas” o “pendientes”**.
5. Calcular y mostrar en pantalla el **porcentaje de tareas completadas** respecto al total.
6. **Aplicar principios de diseño web**: diferenciar visualmente las tareas hechas (ej. en verde o tachadas) y pendientes (ej. en rojo), cuidando tipografía y colores para mayor claridad.

#### 5. Simulador de tienda online sencilla

**Requisitos funcionales:**

1. Mostrar un **catálogo de productos** con nombre y precio (mínimo 5 productos).
2. Permitir que el usuario **añada productos al carrito de compra**.
3. Mostrar el contenido del **carrito actualizado** y el total acumulado.
4. Calcular el precio final y aplicar un **descuento automático** si el importe supera una cantidad establecida (ej. 100 €).
5. Permitir **eliminar productos del carrito** antes de finalizar la compra.
6. **Aplicar principios de diseño web**: diferenciar claramente el catálogo del carrito mediante colores adecuados y tipografía coherente para simular un entorno de compra real.

## Requisitos de Entrega

El alumnado entregará en **Moodle** un archivo comprimido en formato **.zip** que contendrá:

* **El código del proyecto**, bien organizado dentro de un directorio específico para el desarrollo de la aplicación.
* El **código fuente deberá estar escrito en inglés**, salvo los comentarios en el código y los mensajes mostrados al usuario o en consola para el programador, que estarán redactados en español para facilitar su comprensión por todo el grupo.
* **Un documento en formato PDF y Word** con la siguiente información:
  + **Portada** con el título del proyecto y la información relevante del grupo (nombre de los integrantes, curso y módulo).
  + **Explicación detallada** de cómo se ha resuelto la actividad, indicando qué elementos de JavaScript se han utilizado (variables, operadores, condicionales, bucles, funciones, etc.) y cómo se han aplicado en el proyecto.
  + **Justificación del diseño visual** empleado (colores y tipografía), explicando brevemente por qué se han elegido y cómo mejoran la legibilidad y la coherencia de la aplicación.
  + **Organización del trabajo en equipo**, explicando la distribución de tareas y cómo se ha gestionado la colaboración.
  + **Enlace al repositorio de GitHub** exclusivo para esta actividad, de manera que pueda comprobarse cómo se ha utilizado el control de versiones durante el desarrollo.
* **(Opcional, valorable positivamente):**
  + Uso de **Jira** para organizar tareas y gestionar la resolución de incidencias. En este caso, se podrá incluir en el documento capturas de la planificación y gestión de tareas.
  + Uso de **GitHub** para el control de versiones tanto de código como de documentación y otros elementos relevantes que se puedan gestionar en el proyecto.

Finalmente, esta actividad será **presentada en clase** ante el resto de compañeros, siguiendo las pautas descritas en el enunciado.

## Checklist de entrega (para el alumnado)

*Esta checklist no forma parte de la evaluación, es únicamente una guía para que el alumnado compruebe que no olvida ningún elemento obligatorio antes de entregar la actividad.*

Antes de entregar la actividad, cada grupo deberá comprobar que ha incluido:

* **Código del proyecto** bien organizado dentro de un directorio específico.
* **Documento en Word y PDF** con:
  + Portada con el título del proyecto y los datos del grupo (nombres de los integrantes, curso y módulo).
  + Explicación detallada de cómo se ha resuelto la actividad, indicando qué elementos de JavaScript (variables, operadores, condicionales, bucles, funciones, etc.) se han utilizado y cómo.
  + Justificación del **diseño visual** (colores y tipografía elegidos), explicando brevemente su selección y coherencia.
  + Descripción de la **organización del trabajo en equipo** (reparto de tareas, dificultades y soluciones).
* **Enlace al repositorio GitHub** exclusivo de la actividad, con el histórico de commits que evidencie el uso del control de versiones.
* *(Opcional, valorable positivamente)* Capturas o evidencias del uso de **Jira** para la planificación y gestión de tareas.
* Preparación de la **presentación en clase**, siguiendo las pautas del enunciado (funcionalidad final, estructura, organización, funcionamiento del código, decisiones de diseño, uso de herramientas).