

# Unidad 1. Arquitecturas y lenguajes de programación en clientes web. Javascript.

## Tarea obligatoria

### Notas importantes:

- Intenta que todas las respuestas tengan tu autoría. No te limites a copiar de Internet o de los compañeros.
- En Indica siempre las fuentes consultadas (enlaces, referencias escritas a los documentos, etcétera).

### Actividad 1. Arquitecturas y lenguajes de programación en clientes web.

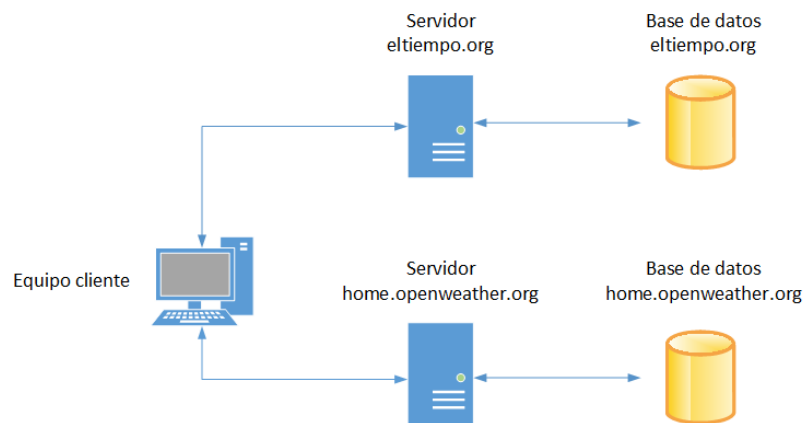
Realiza los siguientes apartados en un documento de texto:

- a) Supongamos que la página web y el resto de archivos que se suministra en la actividad (**actividad1a.zip**) está alojada en un servidor web <https://eltiempo.org>. Cuando se visita esta página web, se muestra un formulario en el que se introduce una ciudad y un país y devolverá los datos del clima actuales. Para llevar a cabo esta operación, el formulario realiza una consulta a una API web que se encuentra en <https://home.openweathermap.org/> (esta parte ya sí ocurre de verdad). Es un servicio web que permite realizar peticiones con los datos indicados más arriba y devolverá en formato **JSON** los datos del clima de la zona indicada. En cuanto al formulario de la aplicación de la web *eltiempo.org*, se debe aclarar su comportamiento:
1. Los dos campos son obligatorios. Si no se suministran, se mostrará un error.
  2. Si la ciudad no existe en el país indicado, se mostrará un error.
  3. Si los datos suministrados son correctos, se mostrarán los datos básicos del clima para la ciudad indicada.

Teniendo en cuenta la arquitectura básica que se muestra a continuación, rellena la tabla con todos los pasos que se realizan para que un usuario pueda obtener el clima de su ciudad. Debes rellenar los pasos que se ejecutan para los 3 casos indicados más arriba. También deberás indicar las tecnologías implicadas (lenguajes de programación y tecnologías de servidor/cliente) – en el diagrama no se indica ninguna, así que tú debes seleccionar posibles lenguajes y tecnologías que podrían ser utilizados. Por ejemplo, para la base de datos del segundo servidor, indico que estará implicado un servidor *MySQL* en una arquitectura *XAMPP*. Intenta que tu respuesta sea lo más detallada posible.

### Notas importantes:

- El código suministrado está ofuscado para que el alumnado entienda que no se trata de explicar lo que hace el código.
- La primera línea de la primera tabla está rellena a modo de ejemplo.
- Puedes añadir y eliminar las filas que necesites en las tablas que se suministran para el resultado de la actividad.



Operación 1. No se han suministrado los dos campos en el formulario				
Equipo fuente	Tecnologías implicadas en fuente	Operación realizada	Equipo destino	Tecnologías implicadas en destino
Equipo cliente	Firefox (no hace falta poner como tecnología HTTP)	Petición realizada a eltiempo.org	Servidor eltiempo.org	PHP recoge la petición, XAMPP
Servidor eltiempo.org	PHP, XAMPP	Respuesta a la petición, muestra la página	Equipo clientes	Firefox, HTML, CSS y JS
Equipo Cliente	Firefox	Realiza una petición rellenando el formulario	Equipo cliente	Firefox, html, css y js
Equipo cliente	Firefox, html, css y js	Se comprueba en el HTML el código, si se han rellenado los dos campos o no	Equipo clientes	Firefox, HTML, CSS y JS

<b>Operación 2. La ciudad suministrada no existe</b>				
<b>Equipo fuente</b>	<b>Tecnologías implicadas en fuente</b>	<b>Operación realizada</b>	<b>Equipo destino</b>	<b>Tecnologías implicadas en destino</b>
Equipo cliente	Firefox (no hace falta poner como tecnología HTTP)	Petición realizada a eltiempo.org	Servidor eltiempo.org	PHP recoge la petición, XAMPP
Servidor eltiempo.org	PHP, XAMPP	Respuesta a la petición	Equipo clientes	Firefox, HTML, CSS y JS
Equipo Cliente	Firefox	Realiza una petición rellenando el formulario	Servidor openweather.org	PHP recoge la petición
Servidor openweather.org	PHP, XAMPP	Recibe la petición con los dos campos rellenos	Base de datos openweather.org	PHP enlazado con MySQL, por lo tanto busca.
Base de datos openweather.org	PHP enlazado con MySQL, por lo tanto busca.	Manda respuesta de lo que ha buscado, tras buscar en la bbdd la ciudad	Servidor openweather.org	PHP, XAMPP
Servidor openweather.org	PHP, XAMPP	Manda el error que se ha encontrado	Equipo clientes	Firefox, HTML, CSS y JS + JSON
Equipo clientes	Firefox, HTML, CSS y JS + JSON	Muestra el error	Equipo clientes	Firefox, HTML, CSS y JS + JSON

Operación 3. Los dos datos se suministran correctamente				
Equipo fuente	Tecnologías implicadas en fuente	Operación realizada	Equipo destino	Tecnologías implicadas en destino
Equipo cliente	Firefox (no hace falta poner como tecnología HTTP)	Petición realizada a eltiempo.org	Servidor eltiempo.org	PHP recoge la petición, XAMPP
Servidor eltiempo.org	PHP, XAMPP	Respuesta a la petición	Equipo clientes	Firefox, HTML, CSS y JS
Equipo Cliente	Firefox	Realiza una petición rellenando el formulario	Servidor openweather.org	PHP recoge la petición
Servidor openweather.org	PHP, XAMPP	Recibe la petición con los dos campos rellenos	Base de datos openweather.org	PHP enlazado con MySQL, por lo tanto busca.
Base de datos openweather.org	PHP enlazado con MySQL, por lo tanto busca.	Con los dos campos correctos, envía el JSON de dicha ciudad al servidor. Solo manda el JSON	Servidor openweather.org	PHP, XAMPP
Servidor openweather.org	PHP, XAMPP	Muestra el resultado de la consulta, cogemos el HTML, css y js para implementarlo con el json que tenía	Equipo clientes	Firefox, HTML, CSS y JS + JSON
Equipo clientes	Firefox, HTML, CSS y JS + JSON	Interpreta el html, css, js y json para mostrarlo por el cliente	Equipo clientes	Firefox, HTML, CSS y JS + JSON

- b) En el apartado anterior debes haber indicado lenguajes usados en el lado cliente. Indica la función que realiza cada uno de ellos dentro de una aplicación web. Indica al menos otro lenguaje de script utilizado para el desarrollo en el lado cliente (no es necesario que se ejecute en el navegador). Señala algunas diferencias (diferencias, ventajas, desventajas, en qué casos es mejor usar uno u otro, etcétera) del lenguaje indicado respecto al lenguaje de script que se ejecuta de forma nativa en los navegadores.

Son esos lenguajes que se pueden dirigir por el navegador y no necesitan conectarse a un servidor ni base de datos exterior.

Por el lado del HTML, se utiliza para estructurar y desplegar una página web y sus contenidos.

En cuanto al CSS, lo usamos para aplicar estilos y diseño.

Si nos referimos a JS, son un conjunto de instrucciones que realizan tareas, con valores, funciones, etc. Permite mejorar la interfaz del usuario haciéndola dinámica.

### Typescript

TypeScript es un lenguaje orientado a objetos y se puede usar desde cualquier navegador, tiene un uso parecido a JS, algo muy útil que incluye es que puede hacer uso de las bibliotecas de JS. Es un lenguaje que se caracteriza debido a que tiene la capacidad de manejar una diversión de datos diferentes. Por lo tanto es escalable y seguro.

Esto último, es más difícil de conocer en JS, puesto que el tipo de dato no se podía conocer hasta el momento de su ejecución, en cambio, TypeScript si que es estático.

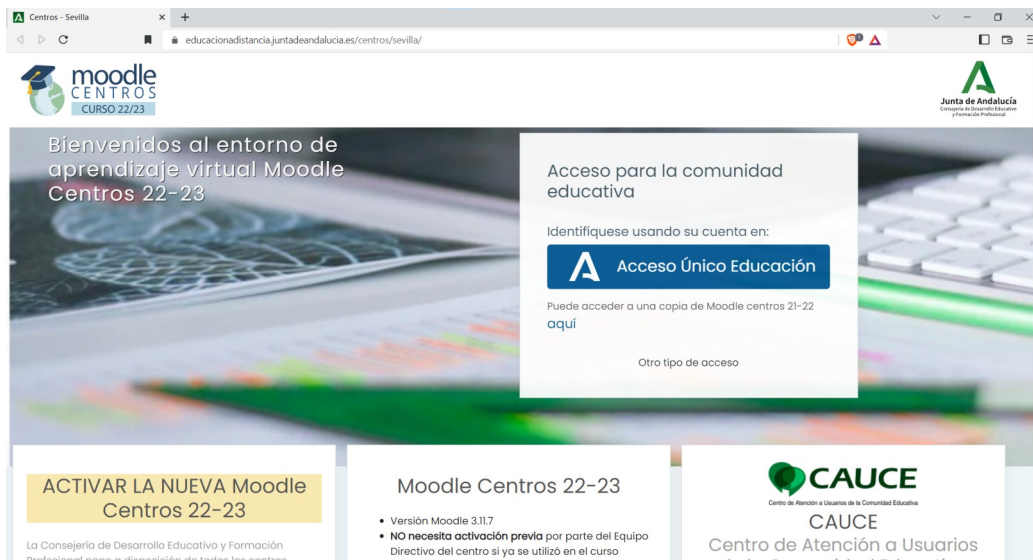
Pulse [aquí](#) para ver la web consultada

- c) Realiza la instalación en Linux de dos navegadores web diferentes a los que tuvieras instalados por defecto (captura de pantalla del navegador ejecutándose). Indica el nombre del motor JavaScript que utiliza cada uno. Añade una captura de pantalla en la que se muestren las herramientas del desarrollador de **ambos navegadores** ejecutando el código de prueba contenido en el proyecto **actividad1c-e.zip**. El código debe aparecer detenido (punto de ruptura) en la primera línea de la función *suma* y la captura debe mostrar la pila de llamadas (*call stack*) y el valor de las variables locales de dicha función. Explica el contenido de la pila de llamada (sólo para un navegador).

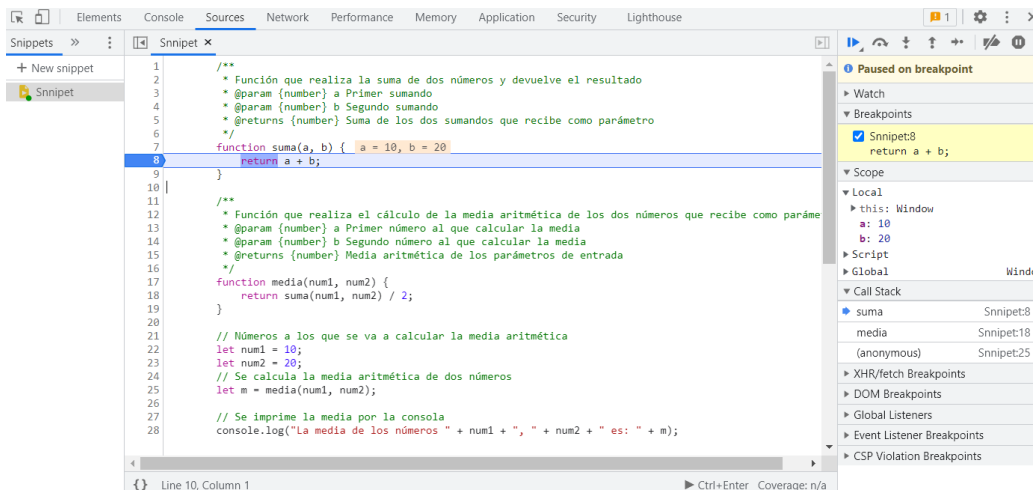
### Respuestas navegador 1:

**Brave**

**Blink V8**



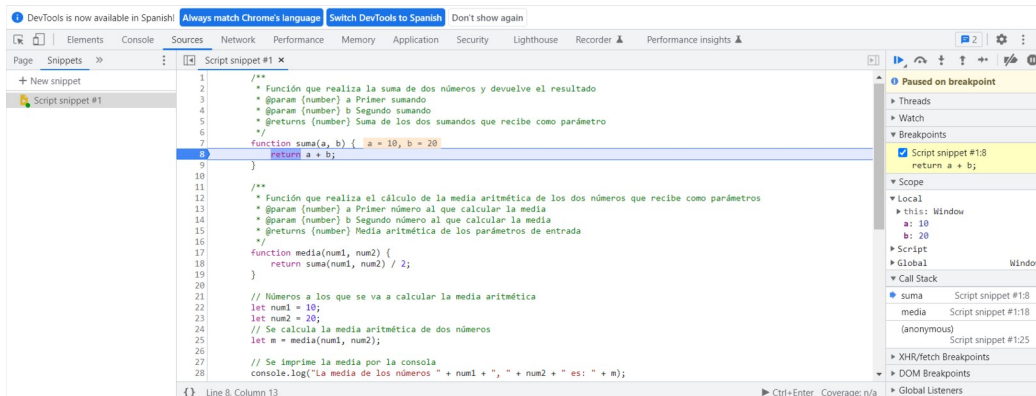
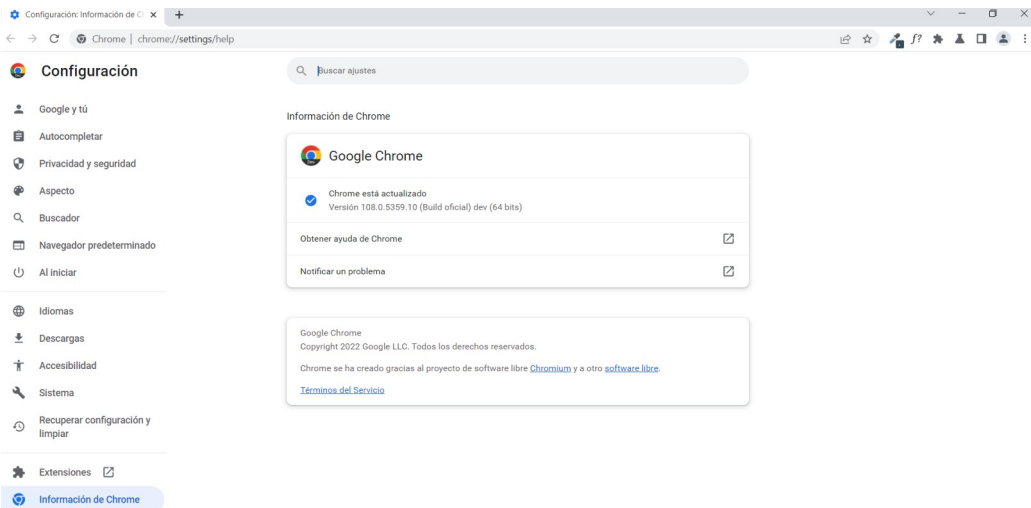
(Podemos ver que es el navegador Brave puesto que vemos el logo del navegador en la derecha de la barra de búsqueda)



## Respuestas navegador 2:

Google Chrome Dev

Blink V8



## Otras respuestas:

En CallStack tenemos los siguientes procesos, en primer lugar suma, puesto que tendremos que hacer el cálculo y devolverlo.

Seguidamente tenemos media puesto que el valor de return de suma va a ser devuelto a la funcion y media y por último anonymous puesto que ya ha acabado el programa.

- d) Indica y explica la estrategia que ofrece el motor de JavaScript de Google Chrome para llevar a cabo la ejecución del código. Indica dos ventajas y dos desventajas de este modelo de ejecución respecto a la estrategia de lenguajes compilados.

Chrome utiliza para JS su propio motor Chrome V8, el motor se encarga de traducir el código a código máquina, de esta manera los ordenadores lo entenderán y luego se ejecuta el código traducido, o compilado. También se encarga de optimizar la ejecución JS

Una de las ventajas es que los lenguajes compilados necesitan una manera de ejecutarse cuando se activan, por lo tanto, para eso sirve el motor V8, puesto que es lo que hace que sin necesidad de servidor se ejecute dicho programa. Este motor es capaz de hilar funciones y ejecutarlas. También, este motor se encarga de almacenar de manera segura las funciones.

Fuente consultada – Click [Aquí](#)

Fuente consultada para las [ventajas](#)

- e) ¿Qué mecanismo de integración de código JavaScript y CSS se ha empleado en el proyecto **actividad1c-e.zip**? Vuelve a montar el proyecto utilizando una estrategia que provoque que la aplicación, en caso de crecer, sea más fácil de mantener. Cambia el nombre de la carpeta del proyecto por **actividad1e** y añádela al entregable de la práctica.

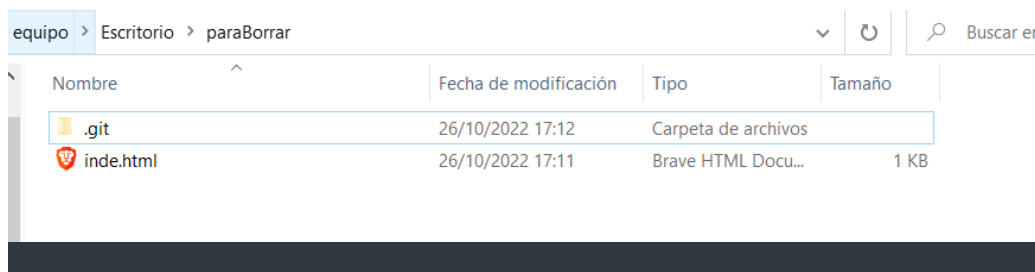
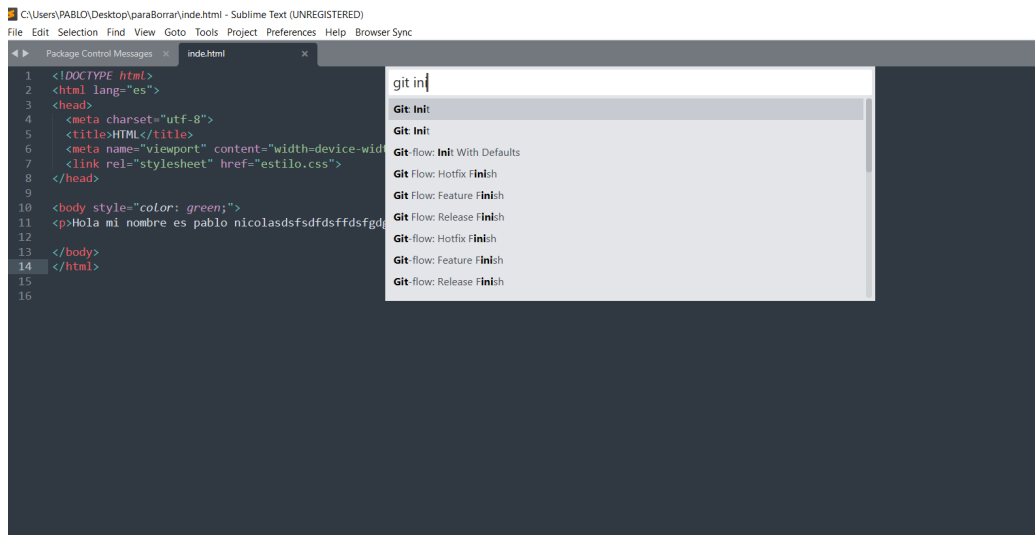
Css lo hemos integrado mediante etiqueta **style**, en cambio, js lo hemos puesto con **script**.

- f) En clase se ha montado un entorno de desarrollo sobre Linux que consta del IDE *Visual Studio Code*, la extensión *Live Server*, el sistema de control de versiones *GIT* y la integración de la depuración desde *Visual Studio Code* con el navegador *Chrome*. Implementa el mismo entorno de desarrollo (con las mismas capacidades) sobre el IDE *Sublime Text*, añadiendo capturas de pantalla que muestren que todas las herramientas se encuentran integradas. Utiliza tu nuevo entorno de desarrollo y realiza una comparativa de su facilidad de configuración e integración, herramientas incluidas que facilitan tu labor (autocompletado y generación automática de código, resaltado y coloreado de sintaxis, verificación de sintaxis, búsqueda de referencias en el código, etc.), etcétera.

```
},
"installed_packages":
[
    "LiveReload",
    "SublimeGit",
    "SimpleReloadPlugin",
    "SimpleRefresh"
]
```

LiveReload, SimpleReloadPlugin y SimpleRefresh son los que he instalado para que se cambie el navegador al guardar los cambios, aparte también he instalado una extensión en Google Chrome para que esto sea posible, adjunto [aquí](#) el tutorial.





Aquí podemos ver como creamos un repositorio git, bastante más sencillo que la práctica anterior.

En cuanto al Web Inspector, no funciona los módulos / plugins que antes si que lo hacían, le han dejado de dar soporte.

En cuanto al autocompletado, es bastante similar a Visual Studio, puesto que desde el primer momento te ayuda, también se parece en el aspecto de autocompletar. Lo he notado muy similar a Visual en cuanto a apariencia y ayuda.

La gran diferencia viene cuando necesitas accesibilidad a los diferentes archivos, también se echa en falta la facilidad de instalar los plugins de Visual y como te lo muestra de manera visual en su barra lateral, en cambio, en Sublime lo tienes que mirar mediante código.

Estas son algunas diferencias, que sin duda me quedaría con Visual por su facilidad de plugins y variedad que incluye, se nota que tiene mucho más soporte y se ve fácilmente los trabajos y todo lo que tienes a mano.

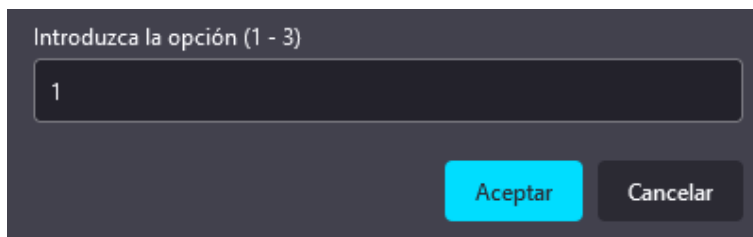
Información propia excepto el vídeo de Live Server.

## Actividad 2. Fundamentos de JavaScript.

Utilizando el proyecto **actividad2** como base, desarrolla una aplicación en JavaScript que implemente un menú que muestre inicialmente por consola la siguiente interfaz:

```
¿Qué desea hacer hoy?  
1. Número par o impar  
2. Saludo  
3. Salir
```

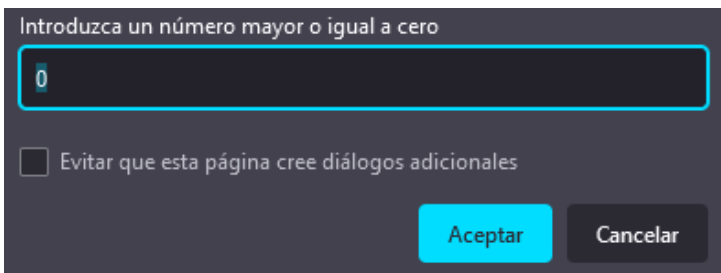
Después debe pedir que se introduzca un número entre 1 y 3 por teclado (el valor por defecto será el 1).



Se deben controlar los siguientes errores mostrando mensajes de error que lo indiquen por consola:

- El valor introducido no es un número.
- El valor introducido es un número pero no está en el rango indicado.

**Opción 1.** Si se selecciona la opción 1, se pedirá al usuario que introduzca un número mayor o igual que cero por teclado (el valor por defecto será 0):

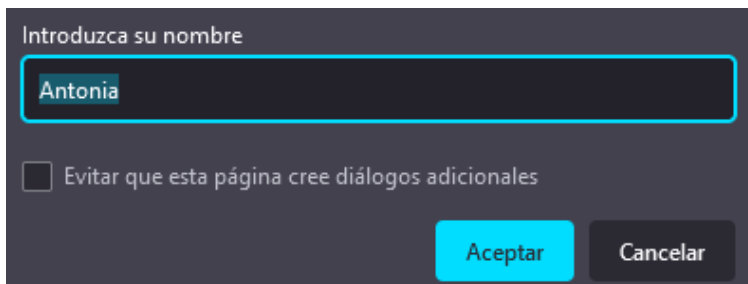


Se deben controlar los siguientes errores mostrando mensajes de error que lo indiquen por consola:

- El valor introducido no es un número.
- El valor introducido no es igual o mayor a cero.

Si se ha introducido un valor correcto, se mostrará un mensaje por consola indicando que el mensaje es par o es impar.

**Opción 2.** Si se selecciona la opción 2, se pedirá al usuario que introduzca un nombre un nombre por teclado (el valor por defecto será *Manuela*):



Se debe controlar el caso en que el valor introducido esté vacío (cadena vacía), mostrando un mensaje de error que lo indique por consola.

Si se ha introducido algo por teclado se debe mostrar el siguiente mensaje:

```
Bienvenido/a <cadena_introducida>
```

**Opción 3.** En caso de haber seleccionado la opción 3 se mostrará el mensaje de despedida siguiente y se terminará la ejecución del programa.

```
Vuelve cuando lo necesites
```

#### Notas importantes:

- Puedes comprobar que en el archivo *index.html* del proyecto **actividad2** aparece un botón que invoca a una función cuando haces *clic*. Esto se hace necesario debido a que usaremos como entrada de datos la función *prompt* y como salida de datos la consola. Si no se hace así, no se podrá abrir la consola hasta debido a que el *prompt* bloquearía su apertura.
- Todos los mensajes que se muestran en la aplicación deben de ser declarados al comienzo del programa con *let* o *const*.
- Elige el tipo de variable más adecuada en cada caso (*let* o *const*).
- Realiza la conversión de datos que sean necesario y lleva a cabo las operaciones entre datos del mismo tipo.
- Comenta tu código de modo que sea más sencillo de mantener.
- Añade a este documento una captura de pantalla en la que se muestre el programa detenido con un punto de ruptura en la línea de código en que se escriba por consola el error producido al introducir un dato que no es un número en la opción 1. Se debe mostrar el valor incorrecto introducido por el usuario en las herramientas del desarrollador (por favor, resalta dicho valor para que sea fácilmente visible).
- **No** se tendrá en cuenta el uso de funciones para hacer el código más modular, aunque se recomienda su uso.

Create and save code snippets for later reuse  
[Learn more](#)

```
65 }
66 }
67
68 function method1(){
69
70     // Creamos variable para comprobar si es true o false
71     let comprobar; comprobar = false
72
73     do {
74         // Solicitamos número y hacemos la comprobación
75         respuesta = prompt("SOLICITUDNUM, DEFAULTVALUE");
76         // En caso de que sea incorrecto devvuelve false y se solicita de nuevo
77         if (isNaN(respuesta) || respuesta < 0){
78             comprobar = false; comprobar = false
79             console.log(NUMEROINVALIDO);
80         } comprobar = true;
81     } while (!comprobar);
82
83     // Aquí veremos si es par o impar
84     if (respuesta % 2 === 0) {
85         console.log(NUMEROPAR);
86     } else {
87         console.log(NUMEROIMPAR);
88     }
89 }
```

▼ Watch

▼ Breakpoints

☒ app.js:79  
console.log(NUMEROINVALIDO);

▼ Scope

► Local

► Script

► Global

► Call Stack

► XHR/fetch Breakpoints

► DOM Breakpoints

▼ Global Listeners

► Event Listener Breakpoints

Aquí podemos ver como el número introducido es -5

## Información de la entrega

En una carpeta con nombre **Actividad\_obligatoria\_UD01** incluye los siguientes elementos:

- Carpeta **actividad1e** con el proyecto del apartado e de la actividad 1.
- Carpeta **actividad2** con el proyecto de la actividad 2.
- Este documento con las actividades rellenas.

Comprime la carpeta completa, teniendo como resultado el archivo **Actividad\_obligatoria\_UD01.zip** y entrégalo en la tarea dentro del tiempo asignado.

## Evaluación de la tarea

### **Criterios de evaluación implicados**

RA1. Selecciona las arquitecturas y tecnologías de programación sobre clientes Web, identificando y analizando las capacidades y características de cada una.

Criterios de evaluación:

- a) Se han caracterizado y diferenciado los modelos de ejecución de código en el servidor y en el cliente Web.
- b) Se han identificado las capacidades y mecanismos de ejecución de código de los navegadores Web.
- c) Se han identificado y caracterizado los principales lenguajes relacionados con la programación de clientes Web.
- d) Se han reconocido las particularidades de la programación de guiones y sus ventajas y desventajas sobre la programación tradicional.
- e) Se han verificado los mecanismos de integración de los lenguajes de marcas con los lenguajes de programación de clientes Web.
- f) Se han reconocido y evaluado las herramientas de programación sobre clientes Web.

RA2. Escribe sentencias simples, aplicando la sintaxis del lenguaje y verificando su ejecución sobre navegadores Web.

Criterios de evaluación:

- a) Se ha seleccionado un lenguaje de programación de clientes Web en función de sus posibilidades.
- b) Se han utilizado los distintos tipos de variables y operadores disponibles en el lenguaje.
- c) Se han identificado los ámbitos de utilización de las variables.
- d) Se han reconocido y comprobado las peculiaridades del lenguaje respecto a las conversiones entre distintos tipos de datos.
- e) Se han añadido comentarios al código.
- f) Se han utilizado mecanismos de decisión en la creación de bloques de sentencias.
- g) Se han utilizado bucles y se ha verificado su funcionamiento.
- h) Se han utilizado herramientas y entornos para facilitar la programación, prueba y depuración del código.

## Valoración y puntuación de la tarea

La tarea se puntúa con un máximo de 10 puntos, teniendo en cuenta los siguientes pesos para cada uno de los elementos de la siguiente rúbrica:

Criterios de evaluación	Ítems evaluables		Peso
RA1.a)	Actividad 1.a)	<ul style="list-style-type: none"> <li>• Se identifican los lenguajes del lado cliente.</li> <li>• Se identifican los lenguajes del lado servidor.</li> <li>• Se identifican las tecnologías implicadas y su interacción.</li> </ul>	2,63%
RA1.c), RA2.a)	Actividad 1.b)	<ul style="list-style-type: none"> <li>• Se describe la función de los lenguajes del lado cliente.</li> <li>• Se indica otro lenguaje del lado cliente.</li> <li>• Se diferencian y clasifican los lenguajes de script correctamente.</li> <li>• La respuesta es personal y se indican las fuentes consultadas.</li> </ul>	5,26%
RA1.b)	Actividad 1.c)	<ul style="list-style-type: none"> <li>• Se realiza la instalación de navegadores web.</li> <li>• Se identifican los motores JavaScript de los navegadores.</li> <li>• Se identifican los mecanismos de ejecución del código mediante la comprensión de la pila de llamadas y de los ámbitos de las variables.</li> </ul>	2,63%
RA1.d)	Actividad 1.d)	<ul style="list-style-type: none"> <li>• Se indica la estrategia empleada por el motor de Google Chrome.</li> <li>• Se indican dos ventajas y dos desventajas de la estrategia de Chrome frente a lenguajes compilados.</li> <li>• La respuesta es personal y se indican las fuentes consultadas.</li> </ul>	5,26%
RA1.e)	Actividad 1.e)	<ul style="list-style-type: none"> <li>• Se indica el mecanismo de integración de código JS y CSS del ejemplo.</li> <li>• Se realiza el cambio de estrategia de integración de modo que el proyecto sea más mantenible.</li> </ul>	5,26%
RA1.f)	Actividad 1.f)	<ul style="list-style-type: none"> <li>• Se instala y demuestra la instalación de las herramientas en Linux.</li> <li>• Se realiza la integración de las herramientas correctamente consiguiendo un entorno de desarrollo funcional.</li> <li>• Se realiza una comparativa de ambos entornos de desarrollo fundamentada en los indicadores expresados en la tarea.</li> </ul>	5,26%
RA2.b)	Actividad 2	<ul style="list-style-type: none"> <li>• Se utilizan diferentes tipos de variables (cadenas, numérico, etcétera y <i>let</i> o <i>const</i>).</li> <li>• Se utilizan los operadores adecuados para cada tipo de dato.</li> </ul>	10,53%
RA2.c)	Actividad 2	<ul style="list-style-type: none"> <li>• Se ha identificado correctamente el ámbito de las variables y usado de forma correcta.</li> <li>• Se declaran las variables en el ámbito en que se usan.</li> </ul>	10,53%
RA2.d)	Actividad 2	<ul style="list-style-type: none"> <li>• Se realizan las conversiones de datos necesarias para el correcto funcionamiento de la aplicación.</li> </ul>	10,53%
RA2.e)	Actividad 2	<ul style="list-style-type: none"> <li>• Se comenta el código de modo que se facilite su comprensión y mantenimiento.</li> </ul>	10,53%
RA2.f)	Actividad 2	<ul style="list-style-type: none"> <li>• Se utilizan adecuadamente las estructuras condicionales en el código, haciendo que la lógica coincida con los requisitos expuestos.</li> <li>• Se tienen en cuenta la eficiencia y mantenibilidad del código.</li> </ul>	10,53%
RA2.g)	Actividad 2	<ul style="list-style-type: none"> <li>• Se utilizan correctamente las estructuras repetitivas en el código, haciendo que la lógica coincida con los requisitos expuestos.</li> <li>• Se tienen en cuenta la eficiencia y mantenibilidad del código.</li> </ul>	10,53%
RA2.h)	Actividad 2	<ul style="list-style-type: none"> <li>• Se utilizan las herramientas de depuración incluidas en los navegadores.</li> <li>• Se depura el programa según los requisitos expuestos.</li> </ul>	10,53%