PLA 5: Fundamentos de JavaScript

Módulo ➜ 2

Dedicación ➜ 5 días (25 horas)

Entrega ➜ 27 de mayo de 2024

Descripción y objetivos

En esta quinta PLA presentaremos las nociones más básicas de JavaScript, para comprobar que todos tienen el nivel suficiente en este lenguaje y así poder luego entender los conceptos más avanzados, que veremos en las siguientes PLA tanto de este módulo como de los posteriores. Esta PLA es, por tanto, una PLA de repaso y de consolidación de conocimientos.

El objetivo de las PLA 5, 6 y 7 es añadir interactividad en una página web para convertirla en una auténtica aplicación web, gracias a la versatilidad que nos ofrece el lenguaje de programación JavaScript.

Recuerden que, en paralelo a la resolución de la PLA, conviene que sigan trabajando en su proyecto final. Próximamente les pediremos muestra de sus progresos en conversaciones individuales con cada uno de ustedes.

Consideración previa

En esta PLA comenzamos de manera oficial con el estudio de JavaScript, que es el primer lenguaje de programación que encontramos, después de haber trabajado con los lenguajes de marcas HTML y de estilos CSS. Por lo tanto, empezamos con la parte de programación que forma el núcleo de este curso y de esta profesión.

Hace ya más de 15 años, Jeff Atwood –uno de los fundadores de la página StackOverflow que seguro conocen muy bien— escribió el artículo "Why Can't Programmers.. Program?", que rápidamente ganó popularidad y que todavía hoy sigue perfectamente vigente. Léalo. Habla de la enorme cantidad de programadores que, de hecho, son incapaces de programar nada.

«Esto no puede ser», pensarán. Pero la realidad es que sí puede ser y hace muchos años que pasa. Antes de la llegada de Internet, la única manera de aprender a programar era... programando muchas horas. Con Internet esto pareció cambiar: poco a poco tuvimos a nuestro alcance una cantidad tan enorme de tutoriales de todo tipo que daba la impresión de que, mirando muchos, por fuerza tenía que ser posible aprender a programar. Y últimamente, con la llegada de herramientas como GitHub Copilot o ChatGPT, parece que incluso podremos programar sin saber programar.

Pero la realidad es que la única manera de aprender a programar sigue siendo la misma de antes: programar mucho. No hay atajos, no hay trucos. Debemos ir encontrándonos, de manera individual, con cada uno de los problemas posibles, para así poder entenderlos y resolverlos. Los tutoriales y los cursos demasiado guiados nos generan un agradable efecto ilusorio de aprendizaje. Pero cuando termina el tutorial o el curso, necesitamos otro, porque todavía no estamos preparados para programar cosas de verdad, solo sabemos reproducir los mismos ejemplos —a menudo bastante básicos— que hemos trabajado repetidamente.

Y con esto entramos en lo que se conoce como tutorial hell (infierno del tutorial), que tal como explican en el artículo "How To Escape Tutorial Hell" consiste en...

Tutorial hell is when you continuously keep on watching tutorials one after another and so on and you feel like, you are learning a lot of things and building some good projects. As soon as you start doing something on your own, you get to know that you don't know anything. [...] Tutorial hell stops you from learning how to solve problems.

En este curso y a partir de esta PLA, por tanto, miraremos de programar mucho. Así intentaremos evitar que cuando termine el curso formen parte del amplísimo grupo de supuestos programadores que, en realidad, no saben programar nada.

Para algunos de ustedes esta PLA resultará muy sencilla y rápida de resolver. Aprovechen el tiempo sobrante para avanzar con toda la parte de HTML y CSS de su proyecto final. También pueden tratar de aprender más CSS (¡nunca llegamos a conocerlo todo!) y profundizar en los métodos de JavaScript que trabajaremos en la última tarea.

Tareas a resolver

Todas las tareas de esta PLA deben resolverse en el archivo pla5.js proporcionado, cada una en el espacio que se indica en el archivo mismo.

En esta PLA no se marcan cuáles son las lecturas recomendadas. Pero, como saben, en MDN tienen una referencia completísima sobre JavaScript. Una de las competencias más importantes que pueden adquirir en su trayectoria inicial como programadores es la capacidad de buscar y encontrar toda la información que necesiten en las diferentes documentaciones oficiales.

Además, también tienen disponible el libro "JavaScript Notes for Professionals" donde, usando el índice o el buscador, encontrarán toda la información que necesiten. Por ejemplo, sobre condiciones en el capítulo 11, sobre arrays en el 12, sobre bucles en el 18... No es necesario que lean todo el libro, por supuesto. Consulten y lean aquellas partes que les hagan falta y luego, para ampliar conocimientos, procuren ir siempre un poco más allá en las lecturas.

Pueden ejecutar el archivo pla5.js en cualquier momento para ver posibles errores sintácticos con el comando node pla5.js ejecutado en su línea de comandos. De esta manera no necesitan enlazar el archivo con ninguna página web.

Siempre que el enunciado dice que hay que mostrar por pantalla (o expresión similar) se está refiriendo a usar console.log() o console.table() para mostrar un resultado en la consola.

Importante: no borren ni modifiquen ninguna línea de código ya existente si no se indica expresamente que deben hacerlo.

Variables de texto, numéricas y booleanas

En el código proporcionado encontrarán definidas una constante de texto y una variable numérica. Creen nuevas variables (con var, let o const según consideren mejor en cada caso) con estas características y en este orden:

cifo debe recoger el texto "CIFO".

pi debe recoger el valor del número pi redondeado a cuatro decimales (lo pueden escribir manualmente, no es necesario que lo recuperen de ningún lado).

distance debe declararse con un valor inicial de 0.

distance debe cambiar, a continuación en la siguiente línea, su valor para almacenar el número de metros que hay en una maratón. Para conseguirlo deben usar el valor de la variable answer, una multiplicación y una suma.

yeap y nope deben tener asignados los valores booleanos verdadero y falso, respectivamente.

condition debe declararse con un valor inicial que refleje si cifo y question tienen el mismo texto.

condition debe cambiar, a continuación en la siguiente línea, su valor para almacenar si distance es mayor o igual que 42000.

pi\_sign debe declararse con el texto "π".

pi\_text debe recoger el texto "The number π is a mathematical constant, approximately equal to 3.1416, defined in..." y debe hacerlo concatenando con + los textos que correspondan y todas las variables definidas previamente que se puedan aprovechar.

cifo\_text debe tener el texto "El Bruc is approximately 42195 meters away from the CIFO location." y debe hacerlo con interpolación (investiguen el concepto javascript string interpolation) de tantas variables definidas previamente como se puedan aprovechar. En la definición de cifo\_text no se puede usar en ningún momento el operador +.

## Estructuras de decisión

En el código proporcionado encontrarán definidas dos constantes.

Escriban, en primer lugar, un bloque if básico que compruebe si age es mayor o igual que 30 y, en caso afirmativo, muestre por consola el mensaje "You are a boomer!".

Escriban, a continuación, un segundo bloque if que muestre por consola el mensaje:

"Too young to be a boomer" si age es 20 o menos.

"Quite close to be a boomer" si age está entre 21 y 35.

"A boomer for sure" si age es 36 o mayor.

Utilicen el mínimo número imprescindible de valores numéricos en este segundo bloque if. Es decir, de entre 20, 21, 35 y 36 utilicen tan pocos como sea posible.

Escriban un tercer bloque switch que en función del valor de la variable lang muestre por consola Spanish si es es, Catalan si es ca, French si es fr y Unknown en cualquier otro caso.

Finalmente, investiguen el concepto javascript ternary operator y úsenlo para escribir una expresión de una única línea (sin ningún punto y coma excepto el del final) de tal manera que si lang es ca muestre por consola Catalan y en caso contrario muestre Some other language.

## 3.1 Estructuras de iteración

En el código proporcionado encontrarán definidas varias variables.

Escriban, en primer lugar, un bucle for (;;) que permita sumar todos los números desde 1 hasta num (incluidos ambos). Esta suma debe quedar recogida, en cada iteración del bucle, en la variable sum. Muestren por consola únicamente el valor final de sum. No muestren ningún resultado intermedio ni ningún texto adicional.

Escriban, en segundo lugar, un bucle while que permita multiplicar todos los números desde 1 hasta num (incluidos ambos). Este producto debe quedar recogido, en cada iteración del bucle, en la variable prod. Muestren por consola únicamente el valor final de prod. No muestren ningún resultado intermedio ni ningún texto adicional.

Finalmente, investiguen el concepto javascript do...while y usen el bucle de este tipo para recorrer los elementos de text\_arr y convertir cada uno de estos elementos en texto en minúsculas. Sustituyan, en cada iteración del bucle, cada elemento en mayúsculas de text\_arr por el texto correspondiente en minúsculas. Una vez terminado el bucle, muestren por consola el array resultante.

Arrays

En el código proporcionado encontrarán definidas varias constantes y variables.

Escriban una función de nombre sumar\_primero\_y\_último que reciba como único parámetro un array de números y devuelva la suma del primer y del último elemento de este array.

Definan el array numbers con los elementos [1, 2, 3, 4, 5] y llamen a la función sumar\_primero\_y\_último pasándole este array como parámetro. Muestren por consola el resultado.

Definan el array coses con los elementos [1, "cosa", true, 2.3, null, "final"] y muestren por consola únicamente los elementos de tipo número contenidos en el array, utilizando para ello el método filter. Muestren por consola el array resultante.

Definan el array language\_codes con los elementos [["es", "español"], ["ca", "catalán"], ["fr", "francés"]] y, a continuación, escriban una expresión que, mediante el uso del método map, muestre por consola únicamente los elementos ["es", "ca", "fr"] sin necesidad de escribir explícitamente estos valores.

Definan el array names con los elementos ["Harry", "John", "Ron", "Jane"]. Investiguen el concepto javascript find y usen este método para encontrar el primer nombre con más de 3 letras. Muestren por consola este nombre.

Investiguen el concepto javascript array includes y usen este método para verificar si el array names contiene a Hermione. Muestren por consola el resultado de la verificación.

Métodos de JavaScript

En el código proporcionado encontrarán definidas varias constantes.

Investiguen y usen el método javascript string startsWith para verificar si story comienza con la palabra Once. Muestren por consola el resultado de la verificación.

Investiguen y usen el método javascript string replaceAll para eliminar, en story, todas las comas y espacios. Muestren por consola el resultado final.

Investiguen y usen el método javascript array slice para copiar, en una nueva variable de nombre subarray, todos los elementos del array elements comprendidos entre los índices 1 y 3, ambos incluidos. Muestren por consola el resultado.

Definan el array numbers2 con los elementos [6, 3, 8, 2, 9, 1, 7, 4] y usen el método javascript array sort para ordenar los elementos del array en orden ascendente. Muestren por consola el array resultante.

Investiguen y usen el método javascript array reduce para sumar los elementos del array numbers2. Muestren por consola el resultado.

Criterios de evaluación

La puntuación de esta actividad formativa se distribuirá de la siguiente manera:

Variables: 2 puntos

Estructuras de decisión: 2 puntos

Estructuras de iteración: 2 puntos

Arrays: 2 puntos

Métodos de JavaScript: 2 puntos

Lecturas recomendadas

Como ya se ha comentado, en esta PLA no se marcan lecturas específicas. El objetivo es que aprendan a buscar y consultar la documentación disponible. Sin embargo, se recomienda la lectura del libro "JavaScript Notes for Professionals" y la consulta de la documentación de MDN.