Preparémonos para trabajar

Como mencionamos anteriormente, **JavaScript se puede utilizar en varios entornos,** aunque lo más habitual es que **sea un navegador web o un servidor con entorno node.js.**

Cada entorno impone una forma ligeramente distinta de utilizar este lenguaje, y aparecen algunos mecanismos o funciones características del mismo.

Aprenderemos a programar usando esta parte central e invariable de JavaScript: cómo declarar variables, escribir funciones, instrucciones condicionales o bucles; todo esto será igualmente utilizable en cualquier entorno en el que decidamos utilizar este lenguaje.

Programar en cualquier lenguaje no es fácil de aprender y es posible que te sientas abrumado por tanta información nueva. Si eres persistente y concentrado, estarás escribiendo scripts simples en poco tiempo, y no hay otra manera de aprender a programar que escribir muchísimo código.

Lo más importante es que no te rindas ni siquiera cuando estés estancado: tómate un descanso, sal a caminar, vuelve a hacerlo con la mente fresca y vuelve a intentarlo. Al final, la carrera se gana con lentitud y constancia.

¡Ahora comencemos!

Herramientas de desarrollo

Como cualquier otra tarea, la programación requiere las herramientas y el espacio de trabajo adecuados. **El desarrollo de software, en la mayoría de los casos, requiere de un editor de código y un compilador o intérprete de un lenguaje determinado.** Este es un conjunto mínimo, que podemos ampliar según sea necesario con otras herramientas.

Además del editor e intérprete de código JavaScript, también podemos utilizar el depurador, que es una herramienta que nos permite, entre otras cosas, pausar el programa en el lugar indicado y analizar su estado actual (p. ej. los valores de las variables indicadas).

En este momento, hay dos opciones. Puede instalar todas las herramientas necesarias en su máquina y trabajar en el entorno local. Este es el enfoque preferido, ya que así es

como se ve en proyectos comerciales reales la mayor parte del tiempo. También puedes personalizar todo para adaptarlo a tus necesidades.

Otro enfoque es utilizar herramientas en línea. Estos pueden ser convenientes, ya que no es necesario instalar ni configurar nada: simplemente funcionan. La mayoría te permiten almacenar tu trabajo en una nube para que puedas acceder a él desde diferentes dispositivos, pero por otro lado, carecen de opciones de personalización y necesitas tener una conexión a Internet constante. Finalmente, podemos pasar a la elección de herramientas.

Entorno de desarrollo en línea

Los entornos en línea, comúnmente conocidos como **áreas de juego de código, son sitios que actúan como un editor simple y un entorno de ejecución**. Todos ellos tienen conjuntos de características similares. **Tienen diferentes interfaces de usuario, pero en principio se comportan de forma similar. Le permiten escribir código, ejecutarlo con fines de prueba y, en la mayoría de los casos, compartirlo con otros usuarios**.

En el caso de JavaScript, donde preparar un entorno local completamente funcional en realidad se reduce a instalar un editor de código y ejecutar el navegador, no son tan importantes como los entornos de desarrollo habituales. Se utilizan principalmente como plataformas de capacitación y prueba, o lugares para publicar soluciones de muestra a problemas de programación.

Entre los programadores de JavaScript, los más populares son los siguientes:

- JSFiddle
- CódigoPen
- JsBin
- plomero

Entorno de desarrollo local

Los requisitos de JavaScript para el entorno de desarrollo en la mayoría de los casos, cuenta con tres elementos: **un editor de código, un intérprete (es decir, un entorno de arranque) y un depurador.**

Dependiendo del nivel de sofisticación, la complejidad del proyecto escrito o el entorno para el que escribimos nuestros programas (del lado del cliente, del lado del servidor, móvil), es posible que también se necesiten otras herramientas.

Estos serán, entre otros:

- **Administradores de paquetes:** permiten la gestión de bibliotecas (que contienen soluciones listas para usar que podemos usar en nuestros programas) o componentes del entorno de desarrollo (por ejemplo, npm o hilo)
- **Ejecutores de tareas y paquetes de módulos:** utilizados, en términos simples, para automatizar el proceso de desarrollo de software y fusionar el código resultante de muchos archivos y bibliotecas (por ejemplo, Grunt o Webpack).
- **Marco de prueba:** permite probar automáticamente la corrección de nuestro programa en busca de posibles errores (por ejemplo, Mocha, Jasmine o Jest)
- **Analizadores de seguridad:** como puede adivinar, se utilizan para controlar la seguridad de nuestra solución (por ejemplo, Snyk, RetireJS u OWASP Dependency Check)