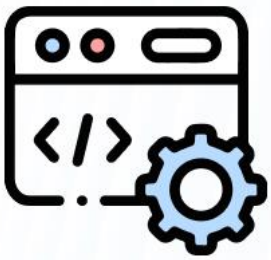
Entorno de desarrollo

Tarea inicial



Manuel Pacheco Sánchez

¿Qué son los entornos de desarrollo?

Un entorno de desarrollo es una herramienta que permite facilitar las tareas del programador con varios servicios integrados.

Básicamente, hace referencia a todo aquello que se necesita a la hora de desarrollar una app o un software. Usualmente se denomina bajo las siglas IDE, consiste en un editor de código integrado con herramientas de construcción y depuradores. En ocasiones, estos sistemas cuentan con muchas funcionalidades avanzadas, como autocompletar código de manera inteligente, lo que facilita de forma exponencial las tareas del desarrollador.

Su objetivo principal es maximizar el rendimiento del programador, ofreciendo servicios integrales en un solo programa. Esto es sumamente útil porque en ocasiones los desarrolladores deben utilizar distintas herramientas para crear proyectos de páginas web, para compilarlos o implementarlos y otras para depurarlos.

El entorno de desarrollo cuenta con una serie de niveles, que son los que permiten que la aplicación o el software que se esté realizando sean de calidad. Estos son necesarios para que el proceso se desarrolle efectivamente. En total, son tres niveles, los cuales se encargan de acciones diferentes y de revisar puntos distintos:

- **Desarrollo**: En este punto lo que se busca es que se realice la prueba del código, el cual debe funcionar sin ningún error, para poder asumir que el proceso se realizó correctamente.

- **Integración**: Cuando se desarrolla una aplicación, esta suele funcionar de forma sincronizada en el servidor ensayo, que es donde se está ensamblando. La prueba final utiliza el servidor de producción, que es al que se va a tener acceso cuando esté lista.

- **Producción**: Una vez se logren superar los pasos anteriores, se considera que no presenta ningún error y está lista para formar parte del servidor.

¿Qué fases identificas en la resolución de problemas?

Cuando aparece un problema en un proyecto de programación, existen diferentes fases que debemos tener en cuenta para que la resolución del problema esté organizada y no nos aparezcan más problemas por no haber tenido cosas en cuenta.

Reconocemos las siguientes fases para llevar a cabo el proceso:

1. **Análisis:** el primer paso para encontrar la solución a un problema es el análisis del mismo. Se debe examinar cuidadosamente el problema a fin de obtener una idea clara sobre lo que solicita y determinar lo que se necesita para conseguirlo. El problema se analiza teniendo en presente la especificación de los requisitos dados por la persona que requiere la solución.

2. **Diseño:** una vez analizado el problema, se diseña una solución que conducirá a un algoritmo que resuelva el problema. Se plantean, la forma en que se reciben los datos de entrada, el proceso por el cual se produce el resultado y la forma en que se muestra la salida. Una vez que se ha terminado de escribir un algoritmo es necesario comprobar que realiza la tarea para la cual fue diseñado y produce el resultado correcto y esperado.

3. **Codificación (implementación)**: el algoritmo diseñado se escribe en la sintaxis del lenguaje de programación seleccionado, obteniendo así el programa fuente.

4. **Ejecución, verificación y depuración:** el programa se ejecuta, se comprueba rigurosamente y se eliminan todos los errores que puedan aparecer (eliminar bugs). Asumiendo una competente codificación, entre mejor se haga todo en las fases de análisis y diseño menos tiempo se gastará buscando, identificando y solucionando errores. Cuando se ejecuta un programa pueden producirse tres tipos de errores: de traducción, de ejecución y lógicos.

5. **Documentación:** documentos que soportan todo lo realizado con respecto al programa. Incluye diseño, codificación, manuales, normas, etc. Se recomienda ir documentando cada cosa que se hace, en vez de esperar hasta el final y tener que documentar todo de principio a fin. Existe documentación interna, incluida dentro del código fuente mediante comentarios, y documentación externa. La documentación es vital para que los usuarios puedan usar el programa y para que los programadores entiendan y modifiquen el código fuente.

6. **Mantenimiento:** el programa se actualiza y modifica cada vez que sea necesario, de modo que se cumplan todas las necesidades adicionales de los usuarios.

¿Asociáis alguna herramienta a cada una de ellas?

Suponiendo que el problema que nos surge es sobre una posible aplicación web, asocio diferentes herramientas para las distintas fases de la resolución del problema.

- Para analizar el problema, que en el caso de nuestra aplicación web, vamos a suponer que no funciona una parte de nuestro código, podemos revisarlo usando **GitHub**, plataforma que incluye una herramienta de revisión de código para los proyectos que alojemos en ella.

- Una vez que hemos detectado el problema, vamos a utilizar **Crucible.** Esta herramienta nos permite que un código pueda ser revisado por un equipo de trabajo simultáneamente, pudiendo proponer cambios, ideas y anotaciones, diseñando de esta forma la solución que se va a realizar.

- Para la codificación, necesitamos un entorno de programación capaz de compilar el programa o la aplicación en la que estamos trabajando y convertir el código en una aplicación funcional. Por ejemplo, podemos usar **Visual Studio** o **Eclipse**.

- Las comprobaciones de ejecución, verificación y depuración utilizamos también entornos de programación, ya que la mayoría de ellos integran funciones que nos permiten hacer nuestras comprobaciones sin tener que utilizar otras aplicaciones externas. De nuevo, vamos a poner como ejemplo el IDE **Crucible**.

- Para la documentación, lo mejor que podemos utilizar es una plataforma donde los mismos usuarios puedan proponer cambios o mejoras a nuestra documentación, de forma qué si hay algún error, se nos haga conscientes de los mismos. Para elaborar nuestra documentación, usaremos **Bit.ai**.

- Para el mantenimiento, es importante que tengamos una plataforma donde los usuarios puedan emitir sus peticiones a futuro de nuestra aplicación web. De esta forma, podemos desarrollar nuevas versiones, y para tener un control total sobre las distintas versiones que utilizamos o que hemos utilizado, lo mejor es usar **Git**. También es recomendable hacer monitorizaciones del rendimiento del servidor que ejecuta nuestra aplicación, ya que nos puede ofrecer mucha información sobre el correcto o no funcionamiento de nuestra app. Nos puede indicar por ejemplo las limitaciones de hardware que puedan aparecer.