INVESTIGA

**¿De qué trabaja un desarrollador de software?**

Un desarrollador de software es un profesional altamente capacitado que se encarga de diseñar, desarrollar y mantener programas informáticos.

**¿Qué empleos puedo aplicar?**

Puedo aplicar a ser desarrollador de páginas web, aplicaciones móviles, videojuegos, ciber seguridad, etc.

**¿Qué habilidades necesito para ser desarrollador de software?**

Se necesitan las siguientes habilidades:

* Saber usar los lenguajes de programación
* Tener conocimientos de la base de datos
* Saber de estructuras de datos y algoritmos
* Tener control de fuentes
* Hacer procedimientos de prueba

Y demás habilidades

**¿Qué es un desarrollador FULLSTACK?**

Es un desarrollador profesional versátil capaz de desenvolverse en un desarrollo web tanto en la parte del cliente como del servidor. Es un perfil muy completo, con amplios conocimientos y herramientas que le permiten afrontar cualquier fase en la construcción de un seftware.

**¿Qué es un desarrollador FRONT-END?**

Es la parte de una web que conecta e interactúa con los usuarios que la visitan. Es la parte que muestra el diseño, los contenidos y la que permite a los visitantes navegar por diferentes páginas.

**¿Qué es BACK-END?**

Es la parte de una web en la que trabajamos y configuramos la web y el interfaz de esta.

**¿Cómo se desarrolla un software?**

Se basa en 7 etapas:

1. Planificación: esto influye directamente en el desarrollo del proyecto y su éxito, establece las fuzzy front end, estas no están sujetas a plazos, en esta fase se incluyen actividades como determinar el ámbito del proyecto, realizar estudios de viabilidad, analizar los riesgos asociados, estimar el coste del proyecto, planificación temporal y asignación de recursos a cada etapa del proyecto.
2. Análisis: se debe indagar sobre que debe hacer exactamente el software, y es justo lo que se hace en esta etapa, descubrir lo que se necesita realmente y llegar a la compresión correcta de los requerimientos del sistema, las características que debe poseer.
3. Diseño: fase donde se estudian las opciones de implementación para el software que se debe crear, se decide también la estructura general de este, es una etapa compleja y debe llevarse a cabo de forma iterativa. Quizás la solución inicial no sea la correcta, pero se debe refinar. Hay catálogos de diseños sencillos que sirven para recoger errores que otros han cometido para evitar caer en lo mismo.
4. Implementación: debes elegir herramientas correctas, un entorno que haga sencillo el trabajo y un lenguaje de programación adecuado para el software que quieras construir. La elección depende de las decisiones de diseño tomadas y del entorno donde va a funcionar el software. Para evitar que el código del programa sea indescifrable debes seguir las pautas que indicamos a continuación:
   * Evita bloques de control no estructurados.
   * Identifica bien las variables y su alcance.
   * Elige algoritmos y estructuras de datos convenientes para el problema.
   * Mantén la lógica de la aplicación lo más sencilla posible.
   * Documenta y comenta apropiadamente el código de los programas.
   * Facilita la interpretación visual del código manejando reglas de formato de código anteriormente acordadas en el equipo de desarrollo.
   * Ten en cuenta la adquisición de recursos obligatorios para que el software funcione, y desarrolla casos de prueba para probar el funcionamiento del mismo de acuerdo a como se vaya programando.
5. Prueba: busca detectar errores cometidos en las etapas previas para corregirlos, es ideal hacerlo antes de que el usuario final los consiga, la prueba será un éxito si se detecta algún error.
6. Despliegue: en esta fase se pone el software en funcionamiento, por ello se debe planificar el entorno considerando las dependencias existentes entre los distintos componentes del mismo. Puede haber componentes que funcionen individualmente, pero al combinarlos tengan problemas, siempre usa combinaciones conocidas que no tengan problemas de compatibilidad.
7. Uso y mantenimiento: la fase más importante del ciclo de vida del desarrollo de un software, pues si este no se rompe ni desgasta con el uso, entonces el mantenimiento incluye 3 puntos distintos:
   * Eliminar defectos descubiertos durante su vida útil (mantenimiento correctivo).
   * Adaptarlo a nuevas necesidades (mantenimiento adaptativo).
   * Agregarle nuevas funcionalidades (mantenimiento perfectivo).