1/18/23 - **Lenguajes Orientados a Objetos**

|  |  |
| --- | --- |
| Puntos clave | Notas |
| No te cases con ningún lenguaje. | **Visual Studio Code**, el editor de código multiplataforma de Microsoft es uno de los preferidos por muchos desarrolladores mientras que **Visual Studio** permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno compatible con la plataforma .NET  **Algunos de los lenguajes de programación Orientados a Objetos son:**  **Java**  nació como un lenguaje OOP, su paradigma principal es este. Este lenguaje es muy utilizado para creación a Apps móviles o del lado de servido como lenguaje Backend.  (Extensión: .java)  **PHP**  es un lenguaje pensado para web y es un lenguaje interpretado, esto quiere decir que el servidor es quién interpreta php.  (Extensión: .php)  **Python**  es un lenguaje fácil de usar, y tiene múltiples usos como del lado Web, Server side, Análisis de datos, Machine Learning, ect. Además, es un lenguaje OOP. (Extensión.py)  **JavaScript**  es un lenguaje interpretado, también es un lenguaje OOP pero este está basado en prototipos. Está pensado para web pero podemos trabajar el backend Node.js.  (Extensión: .js)  **C#**  **Ruby**  **Kotlin** |
| Sumario: | |

1/4/23

|  |  |
| --- | --- |
| Puntos clave | Notas |
|  | Los lenguajes orientados a objetos basados en clases, como Java y C++, se basan en el concepto de dos entidades distintas: **clases e instancias**.  **Una clase** define todas las propiedades (considerando como propiedades los métodos y campos de Java, o los miembros de C++) que caracterizan un determinado conjunto de objetos. Una clase es una entidad abstracta, más que cualquier miembro en particular del conjunto de objetos que describe.  Por ejemplo, la clase Empleado puede representar al conjunto de todos los empleados.  **Una instancia**, por otro lado, es la instanciación de una clase; es decir, uno de sus miembros. Por ejemplo, Victoria podría ser una instancia de la clase Empleado, representando a un individuo en particular como un empleado. Una instancia tiene exactamente las mismas propiedades de su clase padre (ni más, ni menos).  **Un lenguaje basado en prototipos, como JavaScript**, no hace esta distinción: simplemente tiene objetos. Un lenguaje basado en prototipos toma el concepto de objeto prototípico, un objeto que se utiliza como una plantilla a partir de la cual se obtiene el conjunto inicial de propiedades de un nuevo objeto. Cualquier objeto puede especificar sus propias propiedades, ya sea cuando es creado o en tiempo de ejecución. Adicionalmente, cualquier objeto puede ser utilizado como el prototipo de otro objeto, permitiendo al segundo objeto compartir las propiedades del primero.  **JavaScript** es un lenguaje orientado a objetos basado en prototipos en lugar de clases. Debido a esta diferencia, puede ser menos evidente cómo JavaScript te permite crear jerarquías de objetos y herencia de propiedades y sus valores. |
| Sumario: | |