CI346 – Programming Languages Comparison

Programmers today have more programming languages at their disposal than ever before and with so much choice comes the importance of knowing what sets these languages apart from one another. There are many important aspects to consider when comparing programming languages; some of which affect how the code is written, such as the paradigm of the language, and others that are less noticeable until the code is run such as whether it is a compiled or an interpreted language. Despite their names, Java and JavaScript are two languages that offer an abundance of differences when compared.

The first difference to look at between the two languages is how the code gets executed. Defining a language as either interpreted or compiled challenging as all programming languages can theoretically be executed using both methods. However, languages are typically implemented primarily using one of the execution methods which means that although it may not be a core feature of the language, it is still a valid point on which languages can be compared.

JavaScript is primarily an interpreted language meaning code is executed one line at a time by an interpreter program. One of the biggest advantages of being an interpreted language is platform independence, programs can easily be run on any platform regardless of hardware as the program is being executed by the interpreter rather than directly by the CPU. Being platform independent means that programs written in an interpreted language can easily be distributed which makes JavaScript an ideal language for web-based applications.

Java is harder to define in the sense of whether it is interpreted or compiled as it falls into both categories even more so than most languages. Whereas other compiled languages will compile code directly to machine code, Java code is firstly compiled into Java Bytecode which is in turn interpreted by the Java Virtual Machine (JVM). Because of this, Java has the same benefit of platform independence as JavaScript does, providing the platform it is running on is Java enabled.

The execution method of a language also ties in to how it is type checked. Java is a strong and statically typed language meaning that the type of a variable is known at compile time and once variables have been declared as a certain type, they cannot be assigned a value of a different (see appendix [1]). The advantages associated with being a statically typed language are that type errors, such as trying to use the ‘+’ operator between an integer and a string, get caught at compile time rather than run time. This leads to a more efficient development cycle as less time is spent tracking down and fixing type errors. Statically typed languages also have the advantage of being compiled into better optimised machine code that can be executed more quickly as the compiler already knows the data type of all variables being used.

JavaScript, on the other hand, lies on the other end of the type checking spectrum being a weak and dynamically typed language. Being weakly typed does not mean that variables in JavaScript do not have a type, it simply means that

**Appendix A**

[1] – Assigning a string to an int variable in Java

int data;  
data = 50;  
data = “Hello World!”; // causes a compilation error

(Bhatnagar, 2018)

**References**

Bhatnagar, M. (2018). *Magic lies here - Statically vs Dynamically Typed Languages*. [online] Medium. Available at: https://android.jlelse.eu/magic-lies-here-statically-typed-vs-dynamically-typed-languages-d151c7f95e2b [Accessed 15 Jan. 2020].