# Java versus JavaScript: a Reflection

Despite the similarity in names and select syntax rules, these two programming languages are fundamentally different at their cores. Their key distinctions can be seen within their typing, class / object definitions, error / exception handling and/or type safety, as well as their usage within AI development.

## Typing

Java is a statically-typed language, which means that variables must be known at the time of compilation and means that variable/method types do not have to be checked before runtime. JavaScript, on the other hand, is a dynamically-typed language, which means its variables are not fixed at compile time and can have its type changed up until the point in which it is used in the live program and can be altered between different points when the program checks it. Java is run on a specific Java Virtual Machine when compiled, while JavaScript is interpreted by the browser compiler.

## Object Modeling

Both Java and JavaScript can be considered "Object-Oriented" languages, as they both support ideas of encapsulation of data, inheritance, and polymorphism. However, JavaScript utilizes "prototypes" to define such objects as it allows for more flexibility when programming, which is important in regards to dynamically-typed languages, while Java is considered a "pure" OOP language that only allows such objects to be created via classes as it allows for more structure that is necessary for a statically-typed language. While prototypes are an underlying "feature" of classes, the strictness of class structure makes it less likely to produce instances of type errors that are commonly seen in JavaScript testing of prototype object instances.

## Exception Handling and Type Safety

Given the static nature of the Java language, type safety is maintained by ensuring any methods accessing / altering it will work with a variable of that same type, performing compile-time checks before the program even runs to ensure the correct variable / method type is always used. Exception handling is handled with "try-catch" blocks and throwing exceptions in the unlikely event that a type safety error is not caught by the Java compiler. JavaScript, on the other hand, implements type-checking that occurs later in runtime, and since this increases the risk of type errors, one can use "typeof" or "instanceof" operators that are placed before variables/objects that allows the determination of a variable / object's type at compile time rather than runtime.

## AI Development Capabilities

Java's abilities for strong typing (strictly enforced data type rules and structure) and ease of connection to databases for complex computations makes it a useful language when improving AI scalability (the AI's ability to adjust complexity, size, and/or speed to match necessary requirements.) JavaScript, being created for websites, has been utilized primarily to implement AI within web-based applications with ease-of-use with HTML and CSS. JavaScript is also used for training models real-time alonsgide clients interactiving witha  life interface through its various libraries, which Java has but is not nearly as efficient / useful compared to languages better suited (such as Python.) Overall, Java is most useful in AI development for larger-scale operations, like deploying and management, while JavaScript is most useful in earlier development and better at working alongside a client to work with real-time.