Data Types, Values, and Variables in JavaScript.

Computer programs work by manipulating values, such as the number 3.14 or the text “Hello World.” The kinds of values that can be represented and manipulated in a programming language are known as types, and one of the most fundamental characteristics of a programming language is the set of types it supports.

When a program needs to retain a value for future use, it assigns the value to (or “stores” the value in) a variable. Variables have names, and they allow use of those names in our programs to refer to values. The way that variables work is another fundamental characteristic of any programming language.

JAVASCRIPT TYPES CAN BE DIVIDED INTO TWO CATEGORIES: PRIMITIVE TYPES AND OBJECT TYPES.

JavaScript’s primitive types include **numbers**, strings of text (known as **strings**), and Boolean truth values (known as **booleans**). The special JavaScript values null and undefined are primitive values, but they are not numbers, strings, or booleans.

Each value is typically considered to be the sole member of its own special type. ES6 (ES2015) adds a new special-purpose type, known as Symbol that enables the definition of language extensions without harming backward compatibility. BigInt(ES2020) defines larger integers than the Number type can hold.

JavaScript has dynamic typing, which means we do NOT have to manually define the data ytype of the value stored in a variable. Instead, data types are determined automatically.

Any JavaScript value that is not a number, a string, a boolean, a symbol, a BigInt, null, or undefined is an object. An object (that is, a member of the type object) is a collection of properties where each property has a name and a value (either a primitive value or another object). One very special object is the global object. An ordinary JavaScript object is an unordered collection of named values. The language also defines a special kind of object, known as an array that represents an ordered collection of numbered values. The JavaScript language includes special syntax for working with arrays, and arrays have some special behavior that distinguishes them from ordinary objects. In addition to basic objects and arrays, JavaScript defines a number of other useful object types. A Set object represents a set of values. A Map object represents a mapping from keys to values. Various “typed array” types facilitate operations on arrays of bytes and other binary data. The RegExp type represents textual patterns and enables sophisticated matching, searching, and replacing operations on strings. The Date type represents dates and times and supports rudimentary date arithmetic. Error and its subtypes represent errors that can arise when executing JavaScript code.

JavaScript differs from more static languages in that **functions and classes** are not just part of the language syntax: **they are themselves values** that can be manipulated by JavaScript programs. Like any JavaScript value that is not a primitive value, functions and classes are a specialized kind of object.