Data Types in Javascript :

Following are the 7 primitive data types in javascript:

Numbers
BigInt
String
Boolean
Undefined
Null
Array
Object

Symbol

Javascript Dynamic Typing:

```
var variable = "Hello World!";

//lets print the data type of this variable
console.log(typeof(variable)); //this will print 'string'

//Now let's update the value to a number
variable = 2; //By this line, we update the value of variable to 2


//Now let's print the data type of updated variable
console.log(typeof(variable)); //This will print: 'number', since 2 is a number
```

*We can put any type in a variable.

For example, a variable can at one moment be a string and then store a number:

```
// no error
let message = "hello";
message = 123456;
```

Number:

```
let n = 123;
n = 12.345;
```

==>The number type represents both integer and floating point numbers.
=>There are many operations for numbers,
e.g. multiplication *, division /, addition +, subtraction -, and so on.
=>Besides regular numbers, there are so-called "special numeric values" which also belong to this data ty
pe:
Infinity, -Infinity and NaN.
=>Infinity represents the mathematical Infinity ∞.
It is a special value that's greater than any number.

ex: alert(1 / 0); // Infinity
ex: alert(Infinity); // Infinity
ex: alert("not a number" / 2); // NaN, such division is erroneous

NaN is sticky. Any further mathematical operation on NaN returns NaN:
---

alert(NaN + 1); // NaN
alert(3 * NaN); // NaN
alert("not a number" / 2 - 1); // NaN

BigInt:
---

*In JavaScript, the “number” type cannot safely represent integer values larger than (253-1)
(that’s 9007199254740991), or less than -(253-1) for negatives.
*number” type can store larger integers (up to 1.7976931348623157 * 10308)

ex:

console.log(9007199254740991 + 1); // 9007199254740992
console.log(9007199254740991 + 2); // 9007199254740992

*So to say, all odd integers greater than (253-1) can’t be stored at all in the “number” type.

*BigInt type was recently added to the language to represent integers of arbitrary length.

A BigInt value is created by appending n to the end of an integer:
---->

ex:

// the "n" at the end means it's a BigInt
const bigInt = 1234567890123456789012345678901234567890n;

=>As BigInt numbers are rarely needed, we don’t cover them here,
but devoted them a separate chapter BigInt. Read it when you need such big numbers.

String:
==========>
A string in JavaScript must be surrounded by quotes.

let str = "Hello";
let str2 = 'Single quotes are ok too';
let phrase = `can embed another ${str}`;
In JavaScript, there are 3 types of quotes.

Double quotes: "Hello".
Single quotes: 'Hello'.
Backticks: `Hello`.

==>Double and single quotes are “simple” quotes.

There’s practically no difference between them in JavaScript.

Boolean (logical type):

The boolean type has only two values: true and false.

This type is commonly used to store yes/no values: true means “yes, correct”, and false means “no, incorr ect”.

For instance:

```
let nameFieldChecked = true; // yes, name field is checked
let ageFieldChecked = false; // no, age field is not checked
```

eg:

```
let isGreater = 4 > 1;

alert( isGreater ); // true (the comparison result is "yes")
```

Null:
=========

*The special null value does not belong to any of the types described above.

*It forms a separate type of its own which contains only the null value:
eg:

```
let age = null;
```

*In JavaScript, null is not a “reference to a non-existing object” or a “null pointer” like in some other langua ges.

*It’s just a special value which represents “nothing”, “empty” or “value unknown”.

*The code above states that age is unknown.

undefined:
==========>
==>The special value undefined also stands apart. It makes a type of its own, just like null.

=>The meaning of undefined is “value is not assigned”.

=>If a variable is declared, but not assigned, then its value is undefined:

eg:

```
let age;

alert(age); // shows "undefined"
```

Technically, it is possible to explicitly assign undefined to a variable:

```
let age = 100;

// change the value to undefined
age = undefined;

alert(age); // "undefined"
```

=> Normally,
one uses null to assign an “empty” or “unknown” value to a variable,
while undefined is reserved as a default initial value for unassigned things.

JavaScript typeof operator:

```
console.log(typeof 10); // 'number'
console.log(typeof 'Hello'); // 'string'
console.log(typeof []); // 'object'

console.log(typeof null); // 'object'
console.log(typeof undefined); // 'undefined'
console.log(typeof function(){}); // 'function'
```

===>
number: for numbers of any kind: integer or floating-point, integers are limited by ±(253-1).
bigint: for integer numbers of arbitrary length.
string: for strings. A string may have zero or more characters, there’s no separate single-character type.
boolean :for true/false.
null: for unknown values – a standalone type that has a single value null.
undefined: for unassigned values – a standalone type that has a single value undefined.
symbol: for unique identifiers.
And one non-primitive data type:
object for more complex data structures