Datatypes and ECMA standards

Datatypes and ECMA standards | chai aur #javascript - YouTube

1. What Is ECMAScript?

- **ECMAScript** (often shortened to ES) is the standard specification for JavaScript maintained by TC39 (a committee).
- It's like a rulebook: every browser or JS environment (like Node.js) follows these rules to execute JS consistently.
- Versions are released regularly — e.g., ES5, ES6, ES2020, and so on.

2. JavaScript Data Types

JavaScript has two main categories of data types:

👉 Primitive Types

1. **String** – text in quotes, e.g., `"hello"`.
2. **Number** – includes integers, decimals, and special values like `NaN` (not a number) or `Infinity`.
3. **BigInt** – for very large integers that exceed the safe limit of normal `Number`.
4. **Boolean** – `true` or `false`.
5. **undefined** – a variable declared but not assigned a value.
6. **null** – an intentional absence of value.
7. **Symbol** – unique identifiers useful for private object properties.

👉 Non-Primitive Type

- **Object** – collections of key–value pairs, or more complex types like arrays and functions.

3. Why BigInt and Symbol?

- **BigInt**: JavaScript's normal numbers can only safely go up to around 9 quadrillion. Use BigInt (e.g., `123n`) for integers larger than that.
- **Symbol**: Creates a unique value each time, perfect for properties that shouldn't collide with others.

4. Type Checking

- Use the `typeof` operator to check the type of a value:
  - `typeof "hello"` → `"string"`
  - `typeof 42` → `"number"`
  - `typeof 10n` → `"bigint"`
  - `typeof Symbol("id")` → `"symbol"`
  - `typeof null` → `"object"` (this is a known JavaScript quirk)
  - `typeof undefined` → `"undefined"`

5. Boxing & Wrapper Objects

- Primitives like strings or numbers can behave like objects when using methods (e.g., `"hello".toUpperCase()`).
- JavaScript temporarily wraps the primitive in an object so you can call methods on it.

📚 Study Notes (Copy-Paste Friendly)

```
# JavaScript Data Types & ECMAScript Notes

## 1. ECMAScript (ES)
- Standard specification by TC39.
- Browser/Node follow ES rules (ES5, ES6, ES2020...).

## 2. Primitive Data Types
- String: "text"
- Number: 123, 3.14, NaN, Infinity
```

- BigInt: 123n (for very large integers)
- Boolean: true / false
- undefined: declared but unassigned
- null: intentional empty value
- Symbol: unique identifier for object keys

3. Non-Primitive Type
- Object: key–value collections (arrays, functions, normal objects)

4. typeof Operator
- typeof "hi" → "string"
- typeof 10 → "number"
- typeof 10n → "bigint"
- typeof Symbol() → "symbol"
- typeof null → "object" // JavaScript bug
- typeof undefined → "undefined"

5. Primitive Wrappers (Boxing)
- Primitives can use methods, e.g.:
  - "hello".toUpperCase() → "HELLO"
  - (because JS wraps them into objects)

6. When to Use
- Use String, Number, Boolean for basic values.
- BigInt for huge numbers.
- Symbol for safe, unique object keys.
- Object for grouped data / structures.