**Chapter 5 : Primitive Types**

Primitive types are predefined and are reserved with a keyword.

Javascript has 7 Primitive types like

* Null - null value- none(constructor function)
* Undefined - undefined (value)- none(constructor)
* Number -123 3.14 - Number()
* Bigint - 123n 256n - BigInt()
* String - “Hello” - String()
* Boolean - true false - Boolean()
* Symbol - none

**String**:

typeof “text”; // string

Typeof “JavaScript Grammar”; // string

Typeof “username”+25; //string

Let string=new String(“hello”); // object

Typeof string; // object

Typeof string.valueOf(); // string

**Template Strings:**

Strings defined using the backtick quotes have the special function.

This is used to embe dynamic variable values inside the string.

Let apples=10;

`There are ${apples} apples in the basket.`

**Let json1=’{“a”:1}’;**

**Let json2=`{“a”:1}`;** These two formats are the correct forms of using ‘ and ` with double quotes in json .

Creative use case:

Let result=stmt?value1:value2;

**Symbol:**

The symbol primitive provides a way to define a completely unique key. It **doesnot have a constructor** and **cannot be initialized using new** keyword.

**Let sym=new Symbol(‘sym’); // error**

It should be like

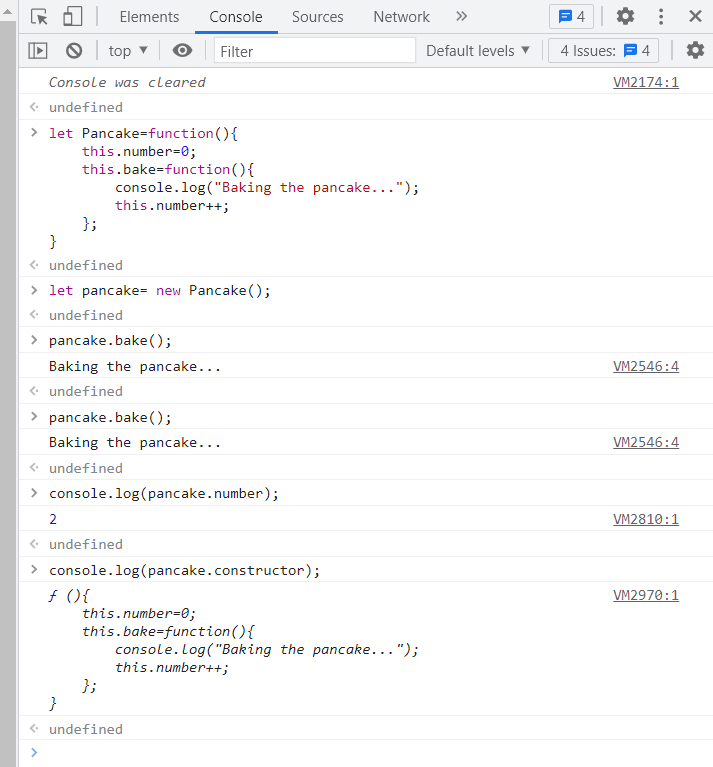
Let sym=Symbol(‘sym’); // symbol created

Symbol(‘sym’)===Symbol(‘sym’); // false

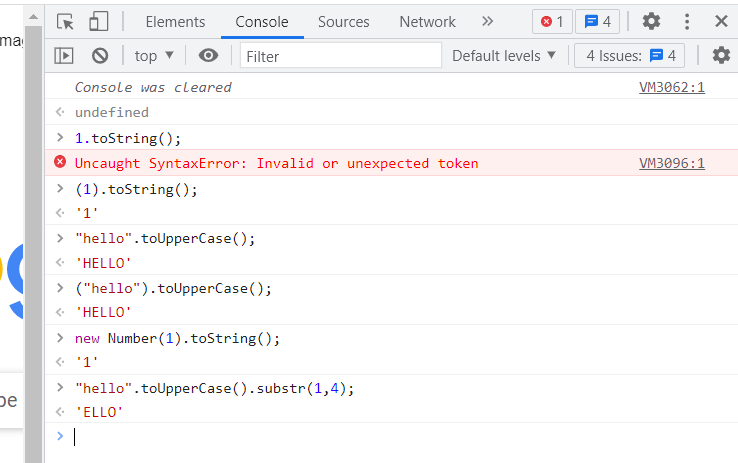
Because every time Symbol(‘sym’) is called a uniques symbol is created and it cannot be equated.

**Constructor and Instances:**

The constructor is only the design of the object type. To use the constructor we need to instantiate it.



**Executing methods on primitive types:**

****