|  |  |
| --- | --- |
| Java | Javascript |
| 1 public class testMain {  2 static double varGlobal1;  3 static StringBuilder varGlobal2;  4 static int variable = 1;  5 **public static void main**(String[] args) {  6 varGlobal2 = new StringBuilder("asd");  7 double varLocal1;  8 int j = 3;  9 double variable = 4.0;  10 varLocal1 = 2;  11 varGlobal1 = 1.0;  12 System.out.println("variables: "+ varGlobal1 +" "+ varGlobal2 +" "+variable +" "+j);  13 }  14} | 1 var variableGlobal;  2 console.log("variableGlobal linea2: "+ variableGlobal);  3 variableGlobal = 1;  4 **function unaFuncion()** {  5 console.log("variableLocal linea5: "+variableLocal);  6 if (true) {  7 let variableBloque = new Date(); // Variable de bloque  8 variableGlobal2 =true; //variable global declarado dentro de una funcion  9 }  10 var variableLocal = new Date(); // Variable local (dentro de la función)  11 console.log("variableLocal linea11: "+variableLocal);  12 }  13 unaFuncion();  14 console.log("variableGlobal1 linea 14: "+variableGlobal);  15 console.log("variableGlobal2 linea 15: "+variableGlobal2); |

Javascript

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identificador | Tipo | r-valor | Alcance | Tiempo de vida |
| varGlobal | Automática | Undefined | 1-15 | 1-15 |
| varGlobal2 | Automática | Undefined | 1-15 | 1-15 |
|  |  |  |  |  |
| varLocal^ | Dinámica | [object] | 4-12 | 10-12 |
|  |  |  |  |  |
| varBloque^ | Dinámica | [object] | 8-9 | 6-9 |

Java

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identificador | L-valor | r-valor | Alcance | Tiempo de vida |
| varGlobal1 | Automática | 0.0 | 3 – 14 | 1-14 |
|  |  |  |  |  |
| varGlobal2^ | Dinámica | indefinido | 4-14 | 6-14 |
| variable (línea 4) | Automática | 0.0 | 5-8 | 1-14 |
| VarLocal1 | Automática | indefindo | 10-13 | 5-13 |
| Main() |  |  | 6-13 | 5-13 ¿? |
| variable (línea 9) | Automatica | indefinido | 10-13 | 5-13 |

Javascript

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identificador | Tipo | r-valor | Alcance | Tiempo de vida |
| varGlobal | Automática | Undefined | 1-15 | 1-15 |
| varGlobal2 | Automática | Undefined | 1-15 | 1-15 |
|  |  |  |  |  |
| varLocal^ | Dinámica | ¿?? [object] | 5-12 | 10-12 |
|  |  |  |  |  |
| varBloque^ | Dinámica | ¿?? [object] | 8-9 | 7-9 |

Ambos lenguajes realizan su liberación de memoria mediante un recolector de basura de forma automática. El recolector de basura es un programa que corre por detrás del programa principal que se encarga de identificar los objetos que no se encuentran referenciados y se eliminan de memoria.

[https://felixgerschau.com/javascript-memory-management/#:~:text=JavaScript%20engines%20have%20two%20places,The%20memory%20heap%20and%20stack](https://felixgerschau.com/javascript-memory-management/#:~:text=JavaScript engines have two places,The memory heap and stack).

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Memory_management>

<https://www.geeksforgeeks.org/java-memory-management/>

<https://www.oracle.com/webfolder/technetwork/tutorials/obe/java/gc01/index.html>

<https://developer.mozilla.org/en-US/docs/Glossary/Hoisting>