http://www.cinstitute.co.in
https://youtube.com/tarunsir
+917742817777

Bitwise operators in javascript

Bitwise operators in JavaScript are used to manipulate individual bits of binary numbers. They treat operands as a sequence of 32 bits (zeros and ones) and perform operations on each corresponding bit. Here are the commonly used bitwise operators in JavaScript:

1. Bitwise AND (**&**):
    - The **&** operator compares each bit of two numbers and returns a new number with the bits set to 1 only if both corresponding bits are 1.
    - Example: **5 & 3** returns **1** because the binary representation of **5** is **0101**, and **3** is **0011**. Comparing the bits, we get **0001**, which is **1** in decimal.
2. Bitwise OR (**|**):
    - The **|** operator compares each bit of two numbers and returns a new number with the bits set to 1 if at least one of the corresponding bits is 1.
    - Example: **5 | 3** returns **7** because the binary representation of **5** is **0101**, and **3** is **0011**. Comparing the bits, we get **0111**, which is **7** in decimal.
3. Bitwise XOR (**^**):
    - The **^** operator compares each bit of two numbers and returns a new number with the bits set to 1 only if the corresponding bits are different (one is 0 and the other is 1).
    - Example: **5 ^ 3** returns **6** because the binary representation of **5** is **0101**, and **3** is **0011**. Comparing the bits, we get **0110**, which is **6** in decimal.
4. Bitwise NOT (**~**):
    - The **~** operator performs a bitwise negation, flipping each bit of a number from **0** to **1** and vice versa. It returns the negation of the number plus **1**.
    - Example: **~5** returns **-6** because the binary representation of **5** is **00000000000000000000000000000101**. Flipping the bits, we get **11111111111111111111111111111010**, which is **-6** in two's complement representation.
5. Left Shift (**<<**):
    - The **<<** operator shifts the bits of a number to the left by a specified number of positions. It effectively multiplies the number by 2 raised to the power of the shift amount.

- Example: **5 << 1** returns **10** because shifting the bits of **5** by 1 position to the left gives **10** (**1010** in binary).

6. Right Shift (**>>**):
    - The **>>** operator shifts the bits of a number to the right by a specified number of positions. It effectively divides the number by 2 raised to the power of the shift amount, discarding the fractional part.
    - Example: **5 >> 1** returns **2** because shifting the bits of **5** by 1 position to the right gives **2** (**0010** in binary).
7. Unsigned Right Shift (**>>>**):
    - The **>>>** operator is similar to the right shift (**>>**), but it fills the leftmost positions with zeros, regardless of the sign of the number.
    - Example: **-1 >>> 16** returns **65535** because shifting the bits of **-1** by 1 position to the right gives **65535** (**0000 0000 0000 0000 1111 1111 1111 1111** in binary).

These operators are mainly used in low-level programming, bitwise operations, and some specific mathematical calculations. They might not be commonly used in everyday JavaScript programming, but they can be handy in certain scenarios.