**Operaciones bit a bit (Bitwise)**

**Martínez Reyes José Alejandro**

**Introducción**

Este documento se realiza con la intención de saber la importancia que tienen las operaciones bit a bit, o también conocidas como Bitwise, ya que nosotros como programadores debemos de entender cómo funcionan estas operaciones debido a que es un lenguaje primitivo y lo debemos de comprender muy bien. También veremos algunas de sus propiedades al manejar uno, dos y hasta tres bits al mismo tiempo.

**Resumen**

**¿Qué es?**

 Es una acción primitiva rápida, soportada directamente por los [procesadores](https://es.wikipedia.org/wiki/CPU), **sustancialmente más rápida** que las que se llevan a cabo sobre el valor real de los operandos.

**Tipos de operadores**

&: El operador AND bit a bit compara cada bit de su primer operando con el bit correspondiente de su segundo operando. Si ambos bits son 1, el bit del resultado correspondiente se establece en 1. De lo contrario, el bit del resultado correspondiente se establece en 0.

^: El operador XOR exclusivo bit a bit compara cada bit de su primer operando con el bit correspondiente de su segundo operando. Si un bit es 0 y el otro bit es 1, el bit del resultado correspondiente se establece en 1. De lo contrario, el bit del resultado correspondiente se establece en 0.

|: El operador OR inclusivo bit a bit compara cada bit de su primer operando con el bit correspondiente de su segundo operando. Si uno de los dos bits es 1, el bit del resultado correspondiente se establece en 1. De lo contrario, el bit del resultado correspondiente se establece en 0.

~: El operador NOT o complemento, es una operación unaria que realiza la negación lógica en cada bit, invirtiendo los bits del número, de tal manera que los ceros se convierten en 1 y viceversa:

<< >>: El desplazamiento lógico se usa para mover bits hacia la izquierda o hacia la derecha para colocarlos en la posición adecuada. En programación, el primer operando representa el binario que queremos modificar mientras que el segundo indica las posiciones a desplazar. En los desplazamientos hacia la izquierda, el bit más significativo (más a la izquierda) se pierde, y se le asigna un 0 al menos significativo (el de la derecha). En los desplazamientos a la derecha ocurre lo contrario. Hay que tener en cuenta que los desplazamientos no son rotaciones; los bits que salen por la izquierda se pierden mientras que los que entran por la derecha son rellenados con ceros. Este tipo de desplazamientos se denominan lógicos a diferencia de los cíclicos o rotacionales.

**Específicos de Microsoft**

Los resultados de la operación bit a bit en enteros con signo se definen en la implementación según el estándar de ANSI C. Para el compilador de Microsoft C, las operaciones bit a bit en enteros con signo funcionan igual que las operaciones bit a bit en enteros sin signo.

**Conclusión**

Las operaciones Bitwise no son tan difíciles de usar, pero si son difíciles de comprender cuando no tienes conocimiento alguno, también son muy importantes en la programación ya que las podrías utilizar para saber si un resultado es verdadero o falso, si esta prendido o apagado.

**Bibliografía**

http://www.etnassoft.com/2012/06/06/operadores-bitwise-en-javascript/

https://msdn.microsoft.com/es-mx/library/17zwb64t.aspx