APPENDIX

C Operator Precedence and Associativity

This table shows the precedence and associativity of all the Java operators. The table is divided into groups, and each operator in a group has the same precedence. The groups of operators are arranged from the highest precedence at the top of the table to the lowest precedence at the bottom of the table. For example, the first group of operators shown is:

```
. [] () ++ --
```

This group of operators has the highest precedence of all the operators; and each of these operators has the same precedence.

Operator	Description	Associativity
`.`	membership	left-to-right
`[]`	array subscript	left-to-right
`()`	method argument list	left-to-right
`++`	postfix increment	left-to-right
`--`	postfix decrement	left-to-right
`++`	prefix increment	right-to-left
`--`	prefix decrement	right-to-left
`+`	unary plus	right-to-left
`-`	unary minus	right-to-left
`~`	bitwise complement	right-to-left
`!`	logical NOT	right-to-left
`new`	object creation	right-to-left
`(type)`	cast	right-to-left
`*`	multiplication	left-to-right
`/`	division	left-to-right
`%`	remainder	left-to-right

Operator	**Description**	**Associativity**
`+`	addition	left-to-right
`+`	string concatenation	left-to-right
`-`	subtraction	left-to-right
`<<`	left shift	left-to-right
`>>`	signed right shift	left-to-right
`>>>`	unsigned right shift	left-to-right
`<`	less than	left-to-right
`>`	greater than	left-to-right
`<=`	less than or equal to	left-to-right
`>=`	greater than or equal to	left-to-right
`instanceof`	type comparison	left-to-right
`==`	equal to	left-to-right
`!=`	not equal to	left-to-right
`&`	bitwise AND	left-to-right
`^`	bitwise XOR	left-to-right
`|`	bitwise OR	left-to-right
`&&`	logical AND	left-to-right
`||`	logical OR	left-to-right
`?:`	conditional	right-to-left
`=`	assignment	right-to-left
`+=`	combined assignment	right-to-left
`-=`	combined assignment	right-to-left
`*=`	combined assignment	right-to-left
`/=`	combined assignment	right-to-left
`<<=`	combined assignment	right-to-left
`>>=`	combined assignment	right-to-left
`>>>=`	combined assignment	right-to-left
`&=`	combined assignment	right-to-left
`^=`	combined assignment	right-to-left
`|=`	combined assignment	right-to-left