Operator Precedence

In the following lesson, you will be introduced to which operators in Dart hold precedence over others.

We'll cover the following

Introduction

Operator precedence determines the order with which different parts of the code/expression should be evaluated. For instance, **1 + 1 * 5** would give us **6** rather than **10** as `*` has higher precedence than `+`. If we wanted **10** we could write the expression as **(1 + 1) * 5** as `()` has higher precedence than `*`.

Precedence Table

Below, you'll find the precedence, with operator precedence being highest at the top and getting lower as you come down. Each operator has a higher precedence than the operators in the rows that follow it.

Description	Operator
Unary postfix	`.`, `?.`, `++`, `--`, `[``]`, `()`
Unary prefix	`-`, `!`, `~`, `++`, `--`, `await`
Multiplicative	`*`, `/`, `~/`, `%`
Additive	`+`, `-`
Shift	`<<`, `>>`, `>>>`

Bitwise AND	`&`
Bitwise XOR	`^`
Bitwise OR	`|`
Relational	`<`, `>`, `<=`, `>=`, `as`, `is`, `is!`
Equality	`==`, `!=`
Logical AND	`&&`
Logical Or	`||`
If-null	`??`
Conditional	`?` `:`
Cascade	`..`
Assignment	`=`, `*=`, `/=`, `+=`, `-=`, `&=`, `^=`, etc.

There might be some operators in the list you aren't familiar with, but that's okay; they will be discussed in future chapters. The purpose of this list is to provide you a comprehensive operator precedence order which you can go back to whenever needed.

And on that note, our discussion on operators comes to an end. In the next lesson, you will be challenged to use your knowledge of operator precedence.