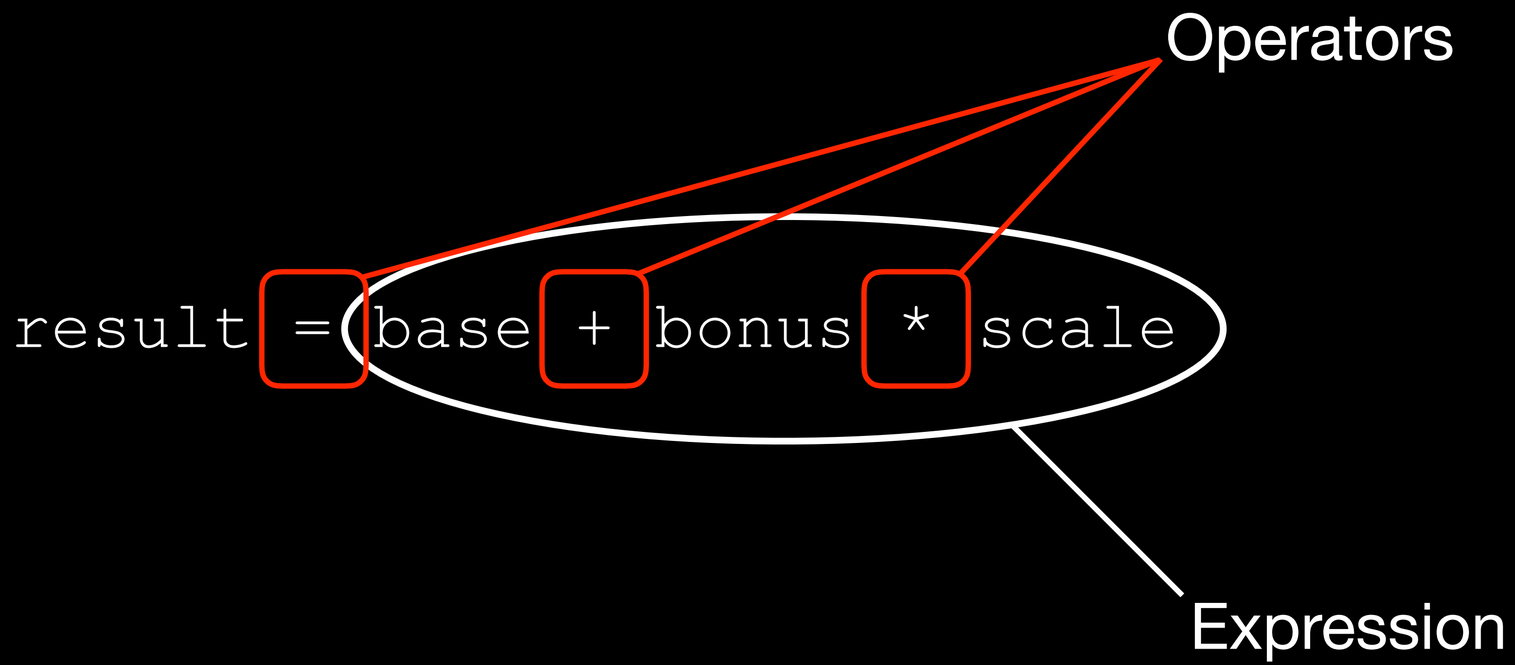
Operators

Expressions and Operators



Unlike other languages (eg C), in Swift the assignment operator does not return a value. As a result, it cannot be used accidentally in a place where the equal to operator (==) was intended.

Operators

An operator is a special symbol or phrase that you use to check, change, or combine values.

Unary

Unary operators operate on a single target. eg. !b

Binary

Binary operators operate on two targets. eg. a + b

Ternary

Ternary operators operate on three targets. eg. a ? b : c

Assignment Operator

The assignment operator initializes or updates the value of a with the value of b.

```
a = b
```

Unlike other languages (eg C), in Swift the assignment operator does not return a value. As a result, it cannot be used accidentally in a place where the equal to operator (==) was intended.

Arithmetic Operators

- Addition +
- Subtraction -
- Multiplication *
- Division /
- Remainder %
- Unary Minus -
- Unary Plus +

Compound assignment operators (eg. “add and assign”)

+=, -=, *=, /=, %=

Comparison Operators

Return a Bool value to indicate whether or not the statement is true.

```
>   greater than
<   less than
>=  greater than or equal to
<=  less than or equal to
==  equality (equal to)
!=  inequality (not equal to)
```

```
1 == 1   // true because 1 is equal to 1
2 != 1   // true because 2 is not equal to 1
2 > 1    // true because 2 is greater than 1
1 < 2    // true because 1 is less than 2
1 >= 1   // true because 1 is greater than or equal to 1
2 <= 1   // false because 2 is not less than or equal to 1
```

How to make useful Boolean Expressions

Here is the most common pattern:

`expression` ***`operator`*** `expression`

Where expression could be a literal value (like 17 or 2.2) or a variable (like playerScore or highScore).
Operator could be a comparison operator such as <, <=, >, >=, ==, != or it could be a logical operator.

```
tin.mouseX < 300
playerScore > highScore
playerLives == 0
```

Conditional Statements

It is often useful to execute different pieces of code based on certain conditions.

If

```
if condition {
   // block of code to be
   // executed only if condition is true
}
```

Conditional Statements

```
var temperature = 30

if temperature <= 32 {
   print(“It’s very cold.”)
}

print(“Have a good day.”)
```

Conditional Statements

```
if condition {
   // block of code to be
   // executed only if condition is true
}
else {
   // block of code to be
   // executed only if condition is false
}
```

Conditional Statements

```
if condition {
   // block of code to be
   // executed only if condition is true
}
else if condition {
   // block of code to be
   // executed only if 2nd condition is true
}
else {
   // block of code to be
   // executed only if all conditions are false
}
```

Logical Operators

Logical operators modify or combine the Boolean logic values true and false.

```
!   Logical NOT
&&  Logical AND
||  Logical OR
```

!

Logical NOT

Inverts the boolean value of an expression.

p	!p
TRUE	FALSE
FALSE	TRUE

& &

Logical AND

Returns true only if both expressions are true.

p	q	p && q
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

||

Logical OR

Returns true only if one or both expressions are true.

p	q	p || q
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

Logical operator examples

```
tin.mouseX < 100 || tin.mouseX > 700

level > 0 && positionY > 1000.0

!soundIsPlaying
```