

Simple Values

let for constants; var for variables

Constants: value doesn't need to be known at compile time, but must assign a value exactly once

Rule for types: constant or variable must have the same type as the value you want to assign it.

Inferring Types: Type can be inferred if value provided when creating constant/var.

Specifying types: Write after the variable, separated by colon

```
let explicitDouble: Double = 70
```

Values are never implicitly converted to another type. If conversion required, must be done explicitly

```
let lottoString = "Today's winning lotto number is "
let lottoNumber = 94
let lottoSentence = lottoString + String(lottoNumber)
```

Backslash string interpolation

```
let friendCount = 2
let friendCountString = "I have \(friendCount) number of friends."
```

Use three double quotation marks for strings that take up multiple lines.

Create arrays and dictionaries using brackets, and access their elements by writing the index or key in brackets.

```
var toys = ["lamborghini", "simba", "furby"]
toys[1] = "fugle r"
toys.append("jellycat")
var bestItem = ["summer fruit": "mango", "winter fruit": "peach"]
```

Arrays grow automatically.

For empty array, write `emptyArray = []`. For an empty dictionary, write `emptyDictionary = [:]`.

If assigning empty array or dict to new var, you need to specify type.

```
let arrayOfGoodCoffees: [String] = []
let dictionaryWithMachines: [String: Float] = [:]
```

Control Flow

Conditionals: if, switch

Loops: for-in, while, repeat - while

Parentheses around condition/loop variable are optional, braces are required

```
let crossAntScores = [1.2, 2.3, 2.2, 4]
for score in crossAntScores {
    if score > 2.5 {
        print("Great cross ant !!")
    }
}
```

In if statement, conditional must be a boolean expression. `if score` is an error, not implicit comparison to zero.

You can assign if conditions to an assignment

```
let scoreDeduction = if teamScore > 10 {
    " "
} else {
    " "
}
print("Score: ", teamScore, scoreDeduction)
```

Optionals: either contains a value or contains nil to indicate a value is missing. A question mark after type of value marks it as optional.

```
var optionalString: String? = "Hello"
print(optionalString == nil)
// prints "false"
```

Use if and let to work with missing values. If optional value is nil, conditional is false and code in braces is skipped. Otherwise, optional value is unwrapped and assigned to constant after let, which makes the unwrapped value available inside block of code.

```
var optionalName: String? = "Big Dog"
var greeting = "Hello!"
if let name = optionalName {
    greeting = "Hello, \(name)"
} else {
    greeting = "Wow!"
}
```

?? operator: If optional value missing, default value used instead

```
let holidayInNov: String? = nil
let holidayInDec: String = "Bali"
let welcomeMsg = "Enjoy \(holidayInNov ?? holidayInDec)"
```



By nixik09

cheatography.com/nixik09/

Not published yet.

Last updated 13th November, 2025.

Page 1 of 1.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>