

# Conditionals

## if Statement

An `if` statement executes a code block when its condition evaluates to `true`. If the condition is `false`, the code block does not execute.

```
var halloween = true

if halloween {
    print("Trick or treat!")
}

// Prints: Trick or treat!
```

## else Statement

An `else` statement is a partner to an `if` statement. When the condition for the `if` statement evaluates to `false`, the code within the body of the `else` will execute.

```
var turbulence = false

if turbulence {
    print("Please stay seated.")
} else {
    print("You may freely move around.")
}

// Prints: You may freely move around.
```

## else if Statement

An `else if` statement provides additional conditions to check for within a standard `if/else` statement. `else if` statements can be chained and exist only after an `if` statement and before an `else`.

```
var weather = "rainy"

if weather == "sunny" {
  print("Grab some sunscreen")
} else if weather == "rainy" {
  print("Grab an umbrella")
} else if weather == "snowing" {
  print("Wear your snow boots")
} else {
  print("Invalid weather")
}
```

// Prints: Grab an umbrella

## Comparison Operators

Comparison operators compare the values of two operands and return a Boolean result:

- `<` less than
- `>` greater than
- `<=` less than or equal to
- `>=` greater than or equal to
- `==` equal to
- `!=` not equal to

```
5 > 1      // true
6 < 10     // true
2 >= 3     // false
3 <= 5     // true
"A" == "a" // false
"B" != "b" // true
```

## Ternary Conditional Operator

The ternary conditional operator, denoted by a `?`, creates a shorter alternative to a standard `if / else` statement. It evaluates a single condition and if `true`, executes the code before the `:`. If the condition is `false`, the code following the `:` is executed.

```
var driverLicense = true

driverLicense ? print("Driver's Seat") :
print("Passenger's Seat")
```

```
// Prints: Driver's Seat
```

## switch Statement

The `switch` statement is a type of conditional used to check the value of an expression against multiple cases. A `CASE` executes when it matches the value of the expression. When there are no matches between the `CASE` statements and the expression, the `default` statement executes.

```
var secondaryColor = "green"

switch secondaryColor {
  case "orange":
    print("Mix of red and yellow")
  case "green":
    print("Mix of blue and yellow")
  case "purple":
    print("Mix of red and blue")
  default:
    print("This might not be a secondary color.")
}
```

```
// Prints: Mix of blue and yellow
```

## switch Statement: Interval Matching

Intervals within a `switch` statement's `case` provide a range of values that are checked against an expression.

```
let year = 1905
var artPeriod: String

switch year {
  case 1860...1885:
    artPeriod = "Impressionism"
  case 1886...1910:
    artPeriod = "Post Impressionism"
  case 1912...1935:
    artPeriod = "Expressionism"
  default:
    artPeriod = "Unknown"
}

// Prints: Post Impressionism
```

## switch Statement: Compound Cases

A compound case within a `switch` statement is a single `case` that contains multiple values. These values are all checked against the `switch` statement's expression and are separated by commas.

```
let service = "Seamless"

switch service {
  case "Uber", "Lyft":
    print("Travel")
  case "DoorDash", "Seamless", "GrubHub":
    print("Restaurant delivery")
  case "Instacart", "FreshDirect":
    print("Grocery delivery")
  default:
    print("Unknown service")
}

// Prints: Restaurant delivery
```

## switch Statement: where Clause

Within a `switch` statement, a `where` clause is used to test additional conditions against an expression.

```
let num = 7

switch num {
  case let x where x % 2 == 0:
    print("\(num) is even")
  case let x where x % 2 == 1:
    print("\(num) is odd")
  default:
    print("\(num) is invalid")
}

// Prints: 7 is odd
```

 Print  Share 