Cheatsheets / **Learn Swift**

Dictionaries

Dictionary

A dictionary is an unordered collection of paired data, or key-value pairs.

```
var dictionaryName = [
  "Key1": "Value1",
  "Key2": "Value2",
  "Key3": "Value3"
]
```

Keys

Every key in a dictionary is unique.
Keys can be be used to access, remove, add, or modify its associated value.

```
// Each key is unique even if they all contain the same value

var fruitStand = [
  "Coconuts": 12,
  "Pineapples": 12,
  "Papaya": 12
]
```

Type Consistency

In a dictionary, the data type of the keys and the values must remain consistent.

```
// Contains only String keys and Int values

var numberOfSides = [
  "triangle": 3,
  "square": 4,
  "rectangle": 4
]
```

Initialize a Populated Dictionary

Dictionary literals contain lists of key-value pairs that are separated by commas; this syntax can be used to create dictionaries that are populated with values.

```
var employeeID = [
  "Hamlet": 1367,
  "Horatio": 8261,
  "Ophelia": 9318
]
```

Initialize an Empty Dictionary

An empty dictionary is a dictionary that contains no key-value pairs.
There is more than one way to initialize an empty dictionary; the method chosen is purely up to preference and makes no impact on the dictionary.

```
// Initializer syntax:
var yearlyFishPopulation = [Int: Int]()

// Empty dictionary literal syntax:
var yearlyBirdPopulation: [Int: Int] =
[:]
```

Adding to a Dictionary

To add a new key-value to a dictionary, use subscript syntax by adding a new key contained within brackets `[ ]` after the name of a dictionary and a new value after the assignment operator (=).

```
var pronunciation = [
  "library": "lai·breh·ree",
  "apple": "a·pl"
]

// New key: "programming", New value:
"prow·gra·muhng"
pronunciation["programming"] =
"prow·gra·muhng"
```

Removing Key-Value Pairs

To remove a key-value pair from a dictionary, set the value of a key to `nil` with subscript syntax or use the `.removeValue()` method.
To remove all the values in a dictionary, append `.removeAll()` to a dictionary.

```
var bookShelf = [
  "Goodnight Moon": "Margaret Wise
Brown",
  "The BFG": "Roald Dahl",
```

```
  "Falling Up": "Shel Silverstein",
  "No, David!": "David Shannon"
]

// Remove value by setting key to nil
bookShelf["The BFG"] = nil

// Remove value using .removeValue()
bookShelf.removeValue(forKey: "Goodnight
Moon")

// Remove all values
bookShelf.removeAll()
```

Modifying Key-Value Pairs

To change the value of a key-value pair, use the `.updateValue()` method or subscript syntax by appending brackets `[ ]` with an existing key inside them to a dictionary's name and then adding an assignment operator (=) followed by the modified value.

```
var change = [
  "Quarter": 0.29,
  "Dime": 0.15,
  "Nickel": 0.05,
  "Penny": 0.01
]

// Change value using subscript syntax
change["Quarter"] = .25

// Change value using .updateValue()
change.updateValue(.10, forKey: "Dime")
```

`.isEmpty` Property

The `.isEmpty` property will return a `true` value if there are no key-value pairs in a dictionary and `false` if the dictionary does contain key-value pairs.

```
var bakery = [String:Int]()

// Check if dictionary is empty
print(bakery.isEmpty)  // Prints true
```

```
bakery["Cupcakes"] = 12

// Check if dictionary is empty
print(bakery.isEmpty)  // Prints false
```

`.count` Property

The `.count` property returns an integer that represents how many key-value pairs are in a dictionary.

```
var fruitStand = [
  "Apples": 12,
  "Bananas": 20,
  "Oranges", 17
]

print(fruitStand.count)  // Prints: 3
```

Assigning a Value to a Variable

To assign the value of a key-value pair to a variable, set the value of a variable to `dictionaryName[keyValue]`.

Note: Assigning the value of a key-value pair to a variable will return an optional value. To extract the value, use optional unwrapping.

```
var primaryHex = [
  "red": "#ff0000",
  "yellow": "#ffff00",
  "blue": "#0000ff",
]

print("The hex code for blue is \
(primaryHex["blue"])")
// Prints: The hex code for blue is
Optional("#0000ff")

if let redHex = primaryHex["red"] {
  print("The hex code for red is \
(redHex)")
}
// Prints: The hex code for red is
#ff0000
```

Iterating Over a Dictionary

A `for - in` loop can be used to iterate through the keys and values of a dictionary.

```
var emojiMeaning = [
  "🤔": "Thinking Face",
  "😪": "Sleepy Face",
  "😵": "Dizzy Face"
]

// Iterate through both keys and values
for (emoji, meaning) in emojiMeaning {
  print("\(emoji) is known as the '\
(meaning) Emoji'")
}

// Iterate only through keys
for emoji in emojiMeaning.keys {
  print(emoji)
}

// Iterate only through values
for meaning in emojiMeaning.values {
  print(meaning)
}
```