Swift 4.20
-Collection Types

—

Ebner

Types

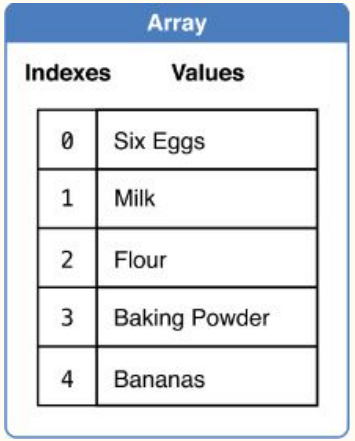
- Arrays
- Sets
- Dictionaries

Arrays

- Werte vom gleichen Typ in geordneter Liste
- Duplikate erlaubt

Syntax:

```
var someInts = [Int]()
print("someInts is of type [Int] with \(someInts.count) items.")
// Prints "someInts is of type [Int] with 0 items."
```



Creating an Array

Array mit Default Value

```
var threeDoubles = Array(repeating: 0.0, count: 3)
// threeDoubles is of type [Double], and equals [0.0, 0.0, 0.0]
```

Literale

```
var shoppingList: [String] = ["Eggs", "Milk"]
// shoppingList has been initialized with two initial items
```

oder

```
var shoppingList = ["Eggs", "Milk"]
```

Accessing and Modifying

Number of Items

```
print("The shopping list contains \(shoppingList.count) items.")
// Prints "The shopping list contains 2 items."
```

Item hinzufügen

```
shoppingList.append("Flour")
// shoppingList now contains 3 items, and someone is making pancakes
```

oder

```
shoppingList += ["Chocolate Spread", "Cheese", "Butter"]
// shoppingList now contains 7 items
```

Accessing and Modifying

Zugriff über Index

```
var firstItem = shoppingList[0]
```

```
shoppingList[0] = "Six eggs"
```

Mehrere Werte ändern

```
shoppingList[4...6] = ["Bananas", "Apples"]
// shoppingList now contains 6 items
```

An bestimmter Stelle einfügen

```
shoppingList.insert("Maple Syrup", at: 0)
// shoppingList now contains 7 items
```

.remove, .isEmpty (boolean)

Iterating

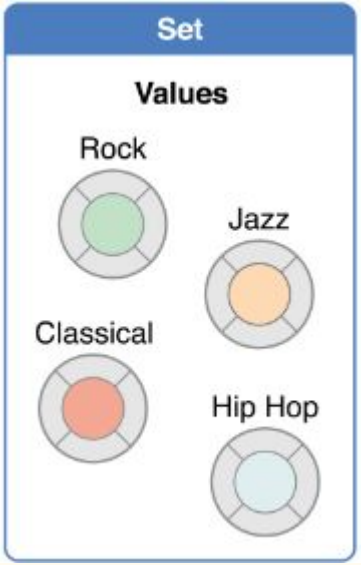
Array mit for-in loop iterieren

```
for (index, value) in shoppingList.enumerated() {
    print("Item \(index + 1): \(value)")
}
// Item 1: Six eggs
// Item 2: Milk
// Item 3: Flour
// Item 4: Baking Powder
// Item 5: Bananas
```

```
for item in shoppingList {
    print(item)
}
```

Sets

- Werte vom gleichen Typ mit undefinierter Ordnung
- Keine Duplikate erlaubt
- Wert muss hashable sein
- Swifts Basic Types (wie String, Int, Double, Bool) sind standardmäßig hashable



Creating a Set

```
var letters = Set<Character>()
```

```
var favoriteGenres: Set<String> = ["Rock", "Classical", "Hip hop"]
var favoriteGenres: Set = ["Rock", "Classical", "Hip hop"]
```

Accessing and Modifying

```
favoriteGenres.insert("Jazz")
```

```
let removedGenre = favoriteGenres.remove("Rock")
```

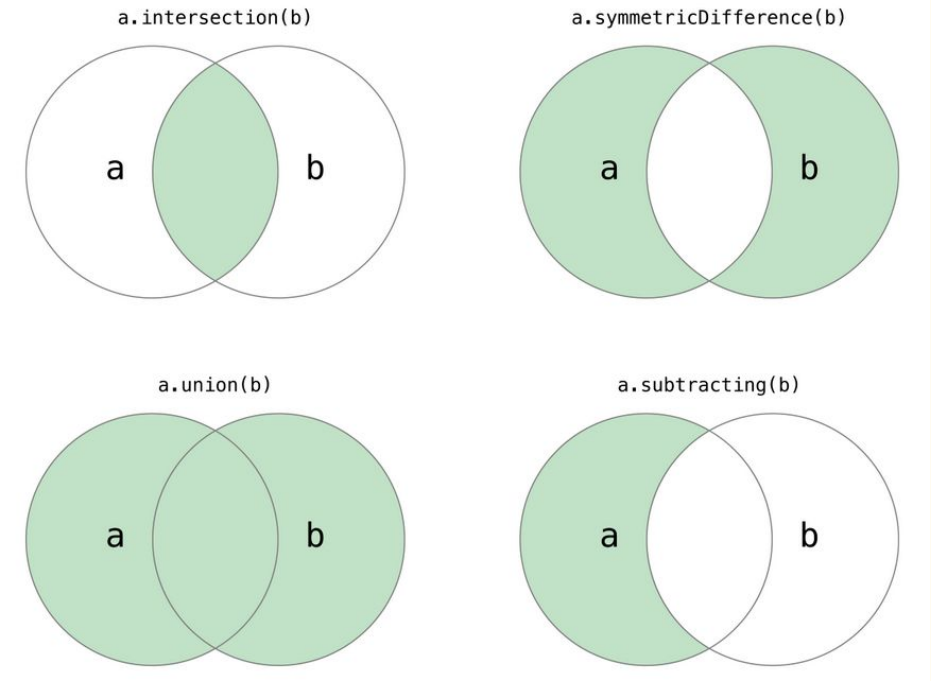
```
favoriteGenres.contains("Funk")
```

.count, .removeAll, .isEmpty

Iterating

```
for genre in favoriteGenres.sorted() {
    print("\(genre)")
}
// Classical
// Hip hop
// Jazz
```

Set Operations

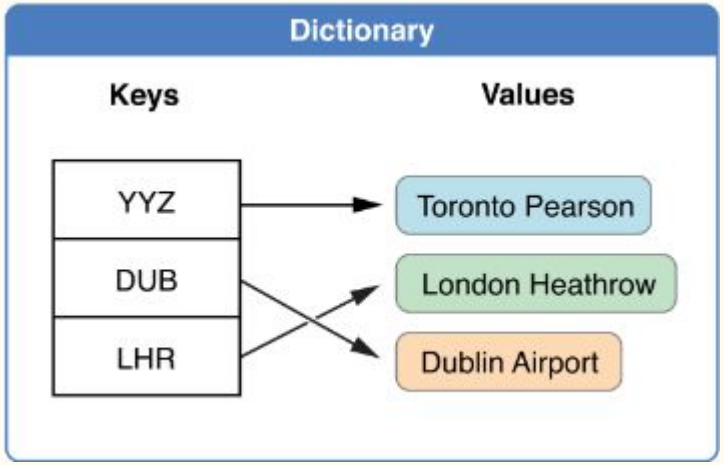



```
let oddDigits: Set = [1, 3, 5, 7, 9]
let evenDigits: Set = [0, 2, 4, 6, 8]
let singleDigitPrimeNumbers: Set = [2, 3, 5, 7]

oddDigits.union(evenDigits).sorted()
// [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Dictionaries

- Key-Value Paare
- undefinierte Ordnung
- Key muss unique sein
- Key muss wie bei Sets hashable sein



Creating a Dictionary

```
var namesOfIntegers = [Int: String]()
namesOfIntegers[16] = "sixteen"
```

```
var airports: [String: String] = ["YYZ": "Toronto Pearson", "DUB": "Dublin"]
```

Accessing and Modifying

```
airports["LHR"] = "London"
```

```
airports["LHR"] = "London Heathrow"
```

```
let oldValue = airports.updateValue("Dublin Airport", forKey: "DUB")
```

.count, .isEmtpy

Iterating

```
for (airportCode, airportName) in airports {
    print("\(airportCode): \(airportName)")
}
// YYZ: Toronto Pearson
// LHR: London Heathrow
```

Task

Rezept 1

- Eier
- Mehl
- Milch
- Salz

sollen in alle 3 Collection Types gespeichert und dann jeweils Ausgegeben werden

Rezept 2

- Eier
- Mehl
- Zucker

Die Zutaten welche in beiden Rezepten vorkommen sollen ermittelt und ausgegeben werden