Swift Filter Map Reduce Sample Code

April 9, 2023

Created By: Debasis Das (Apr 2023)

Swift Filter Examples

Simple Filter Examples

```
let numbers = [1,2,3,4,5,6,7,8,9,10]
let evenNumbers = numbers.filter { $0 % 2 == 0 }
print("evenNumbers = ",evenNumbers)
//evenNumbers =  [2, 4, 6, 8, 10]

let oddNumbers = numbers.filter { $0 % 2 == 1 }
print("oddNumbers = ",oddNumbers)
//oddNumbers =  [1, 3, 5, 7, 9]

//Get number greater than x in an array using filter
let numGreaterThan5 = numbers.filter{ $0 > 5}
print("numGreaterThan5 = ",numGreaterThan5)'
//numGreaterThan5 =  [6, 7, 8, 9, 10]

let names = ["John","Jane","Mary"]
let namesStartingWithJ = names.filter{$0.starts(with: "J")}
let namesContainingO = names.filter{$0.contains("o")}

print("namesStartingWithJ = ",namesStartingWithJ)
//["John", "Jane"]

print("namesContainingO = ",namesContainingO)
["John"]
```

Filtering a Array of Structs

```
struct Person {
  var name: String
  var age: Int
  var isVegan: Bool
}

let people = [
  Person(name: "Alice", age: 25, isVegan: true),
  Person(name: "John", age: 30, isVegan: false),
  Person(name: "Jane", age: 20, isVegan: true),
  Person(name: "Mary", age: 35, isVegan: false),
  Person(name: "Dave", age: 28, isVegan: true)
]

let veganFolks = people.filter { $0.isVegan }
print("veganFolks = ",veganFolks)
//veganFolks =  [__lldb_expr_68.Person(name: "Alice", age: 25, isVegan: true),
__lldb_expr_68.Person(name: "Jane", age: 20, isVegan: true),
__lldb_expr_68.Person(name: "Dave", age: 28, isVegan: true)]

let veganFolksAgeMoreThan25 = people.filter { $0.isVegan && $0.age > 25 }
print("veganFolksAgeMoreThan25 = ",veganFolksAgeMoreThan25)
//veganFolksAgeMoreThan25 =  [__lldb_expr_68.Person(name: "Dave", age: 28, isVegan:
true)]
```

Filtering an array of structs- Example 2

```
let employees = [
    Employee(name: "John", age: 25, hobbies: ["reading", "running", "gaming"]),
    Employee(name: "Jane", age: 32, hobbies: ["writing", "traveling", "painting"]),
    Employee(name: "Mary", age: 42, hobbies: ["reading", "hiking", "fishing"]),
    Employee(name: "David", age: 28, hobbies: ["cooking", "dancing", "photography"]),
    Employee(name: "Anton", age: 37, hobbies: ["gardening", "movies", "music"])
]

let filteredEmployeesWithReadingHobby = employees.filter
{$0.hobbies.contains("reading") }
print("filteredEmployeesWithReadingHobby = ",filteredEmployeesWithReadingHobby)
//filteredEmployeesWithReadingHobby =  [__lldb_expr_68.Employee(name: "John", age:
25, hobbies: ["reading", "running", "gaming"]), __lldb_expr_68.Employee(name: "Mary",
age: 42, hobbies: ["reading", "hiking", "fishing"])]

let filteredEmployeesWithAgeGT30WithReadingHobby = employees.filter { $0.age > 30 &&
$0.hobbies.contains("reading") }
print("filteredEmployeesWithAgeGT30WithReadingHobby= ",
filteredEmployeesWithAgeGT30WithReadingHobby)
//filteredEmployeesWithAgeGT30WithReadingHobby=  [__lldb_expr_68.Employee(name:
"Mary", age: 42, hobbies: ["reading", "hiking", "fishing"])]
```

Swift Map Examples

```swift
let celsiusTemperatures = [0, 10, 20, 30, 40, 50]

let fahrenheitTemperatures = celsiusTemperatures.map { celsius -> Int in
    let fahrenheit = celsius * 9/5 + 32
    return fahrenheit
}
print("celsiusTemperatures = ",celsiusTemperatures)
print("fahrenheitTemperatures = ",fahrenheitTemperatures)

//celsiusTemperatures =  [0, 10, 20, 30, 40, 50]
//fahrenheitTemperatures =  [32, 50, 68, 86, 104, 122]

struct User {
    var firstName: String
    var lastName: String
    var age: Int
}

let users = [
    User(firstName: "John", lastName: "Doe", age: 25),
    User(firstName: "Jane", lastName: "Doe", age: 32),
    User(firstName: "Mary", lastName: "Jane", age: 42),
]

let fullNames = users.map { user -> String in
    return "\(user.firstName) \(user.lastName)"
}

print("fullNames = ",fullNames)
//fullNames =  ["John Doe", "Jane Doe", "Mary Jane"]
```

```
struct Product {
    var name: String
    var price: Double
}

struct Order {
    var products: [Product]
    var total: Double {
        return products.reduce(0.0) { $0 + $1.price }
    }
}

let orders = [
    Order(products: [
        Product(name: "Prod A", price: 799.0),
        Product(name: "Prod B", price: 159.0),
        Product(name: "Prod C", price: 399.0)
    ]),
    Order(products: [
        Product(name: "Prod D", price: 1299.0),
        Product(name: "Prod E", price: 79.0),
        Product(name: "Prod A", price: 49.0)
    ])
]

let productNames = orders.map { order -> [String] in
    let names = order.products.map { $0.name }
    return names
}
print("productNames = ",productNames)
//productNames =  [["Prod A", "Prod B", "Prod C"], ["Prod D", "Prod E", "Prod A"]]

let prodNamesFlattened = productNames.flatMap { $0 }
print("prodNamesFlattened = ",prodNamesFlattened)
//prodNamesFlattened =  ["Prod A", "Prod B", "Prod C", "Prod D", "Prod E", "Prod A"]
```

Swift Reduce Examples

```
//REDUCE
let nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
let sumOfEvens = nums.reduce(0) { (result, number) -> Int in
    return number % 2 == 0 ? result + number : result
}
print("sumOfEvens = ",sumOfEvens)
//sumOfEvens =  30
```

```
struct User {
    var firstName: String
    var lastName: String
    var age: Int
}

let users = [
    User(firstName: "John", lastName: "Doe", age: 25),
    User(firstName: "Jane", lastName: "Doe", age: 32),
    User(firstName: "Mary", lastName: "Jane", age: 42),
]

let totalAge = users.reduce(0) { (result, user) -> Int in
    return result + user.age
}

let averageAge = Double(totalAge) / Double(users.count)

print("averageAge = ",averageAge)
//averageAge =  33.0
```