



USEFUL DATA TYPES

STRING

- A sequence of characters.
- It's actually from `java.lang.String`. It's a Java class.
- A string is immutable.

→ **अमर !**

.

```

object MyString {
  val s1: String = "Hello there. "
  val s2: String = "General Kenobi"
  val n1 = 66;
  val n2 = 98.45

  def main(args: Array[String]): Unit = {
    println(s1.length())
    ① println(s1 + s2)
    ② println(s1.concat(s2))

    → printf("%s: Order (%d) ,has been %f percent completed.", s1, n1,n2)
    val result = printf("%s: Order (%d) ,has been %f percent completed.", s1, n1,n2)
    println(result)
    → println("%s: Order (%d) ,has been %f percent completed.".format(s1, n1,n2))
  }
}

```

→ result คือ "ฟังก์ชัน"

→ ปรินต์สิ่งที่ฟังก์ชันคืนค่า

13

① Hello there. General Kenobi

② Hello there. General Kenobi

→ Hello there. : Order (66) ,has been 98.450000 percent completed. Hello there. : Order (66) ,has been 98.450000 percent completed.()

→ Hello there. : Order (66) ,has been 98.450000 percent completed.

example01 > src > main > scala > Data_Types > MyString.scala

Project

example01 E:\Dropbox\teaching\ProgLangSlides\SCALA\exam

.bsp

.idea

project [example01-build] sources root

src

main

scala

Data_Types

MyString

homework

lecture01_commonfeatures

Lecture02_HigherOrderAndCurrying

Lecture03_List_RecursionSupport

test

scala

target

global-logging

Question07.scala

MyString.scala

```
1 package Data_Types
2
3 object MyString {
4     val s1: String = "Hello there. "
5     val s2: String = "General Kenobi"
6
7     def main(args: Array[String]): Unit = {
8         println(s1.length())
9         println(s1 + s2)
10        println(s1.concat(s2))
11
12        println()
13    }
14 }
```

hover to read description

@NotNull

public String concat(

@NotNull String str

)

Concatenates the specified string to the end of this string.

If the length of the argument string is 0, then this String object is returned. Otherwise, a String object is returned that represents a character sequence that is the concatenation of the character sequence represented by this String object and the character sequence represented by the argument string.

Examples:

"cares".concat("s") returns "caress"

"to".concat("get").concat("her") returns "together"

Params: str – the String that is concatenated to the end of this String.

Returns: a string that represents the concatenation of this object's characters followed by the string argument's characters.

External Method concat:

Run: MyString

E:\Dropbox\Java\jdk17.0.1\bin\java.exe "-javaagent:E:\Dropbox\I

13

Hello there. General Kenobi

Hello there. General Kenobi

Process finished with exit code 0

Version Control

Run

TODO

Problems

Terminal

Build

Dependencies

Build completed successfully in 2 sec, 192 ms (5 minutes ago)

Type here to search

33°C

1:46 PM

3/7/2022

ARRAY

- Store fixed size sequential data (must have the same type)
- Default value for a slot depends on its data type.

```
object MyArray {  
  val a: Array[Int] = new Array[Int](10)  
  var b = Array(1,2,3,4) //initializer list
```

```
def main(args: Array[String]): Unit = {  
  println(a) //will print address
```

0 ถึง a.length - 1

```
for(i <- 0 ≤ .to( ≤ a.length-1)) { // print default values  
  print(a(i) + ", ")
```

```
}  
println("-----")
```

```
for(i <- 0 ≤ .until( < a.length)) { // using "to" -> a.length-1
```

```
  a(i) = i ← assign ค่าใหม่
```

```
}  
for(i <- 0 ≤ .until( < a.length)) {  
  print(a(i) + ", ")
```

```
}  
println("-----")
```

```
for(x <- a){ //for each  
  print(x + ", ")
```

```
}
```

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -----
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -----
0, 1, 2, 3, 4, 5, 6, 7, 8, 9,

ARRAY MAY NEED “IMPORT”

```
import Array._  
Object MyArray02 {  
  
    val ar1 = Array("Luke","Han","Leia")  
    val ar2 = Array("Yugi", "Judai", "Yusei")  
  
    def main(args: Array[String]): Unit = {  
        val c = concat(ar1,ar2)  
        for(x <- c){ //for each  
            print(x + ", ")  
        }  
    }  
}
```

Luke, Han, Leia, Yugi, Judai, Yusei,

SET

- Collection of non-duplicated data.
- They have to have the same data type.
- By default, set is immutable.
- Set is not ordered.
 - So its member does not have index.


```

object MySet {
  val s1: Set[String] = Set("Luke", "Han", "Leia", "Luke") //immutable
  var s2 = scala.collection.mutable.Set("Yugi", "Judai", "Yusei") //mutable

  def main(args: Array[String]): Unit = {
    println(s1)
    println(s1 + "PP") //create a new set
    s2.add("Jojo") //add data to an existing set
    s2.add("Judai")
    println(s2)

    println(s2("Judai")) //Since there is no index, this checks for existence.
    println(s2.head)
    println(s2.tail)
    println(s2.isEmpty)
  }
}

```

เปลี่ยนไม่ได้

เปลี่ยนได้

สร้าง set ใหม่ ที่มีสมาชิกใน s1 และ PP

s2 เปลี่ยนได้

return true ถ้ามีสมาชิกตัวนี้

head กับ tail "ก อก อก อก"

! sequence set ไม่นับ

Set(Luke, Han, Leia)
 Set(Luke, Han, Leia, PP)
 HashSet(Judai, Jojo, Yugi, Yusei)
 true
 Judai
 HashSet(Jojo, Yugi, Yusei)
 false

```
object MySet02 {  
  val s1: Set[String] = Set("Luke", "Han", "Leia", "Luke") //immutable  
  var s2 = scala.collection.mutable.Set("Vader", "Luke", "Chewy", "Han") //mutable  
  
  def main(args: Array[String]): Unit = {  
    println(s1 ++ s2) //union into new set ==> s1.++(s2) → HashSet(Luke, Chewy, Vader, Han, Leia)  
    println(s1 & s2) //intersect into new set ==> s1.intersect(s2) → Set(Luke, Han)  
    println(s1.max) // max value → Luke  
    println(s1.diff(s2)) //difference into new set → Set(Leia)  
    println("-----")  
    s2.foreach(println) //for loop of a set → Chewy  
    println("-----") → Han  
    for(x <- s2){ //normal foreach → Luke  
      println(x) → Vader  
    }  
  }  
}
```

Chewy
Han
Luke
Vader

Chewy
Han
Luke
Vader

MAP

- A collection of (key, value) pairs.
- A key is unique.
- you can choose between mutable/immutable map.

```
object MaMap {
```

```
val mymap: Map[Int, String] = Map(1 -> "Kim", 1 -> "John", 2 -> "Ann", 3 -> "May")
```

มันจะเอา "ตัวล่าสุด"

```
def main(args: Array[String]): Unit = {
```

```
println(mymap) มันพิมพ์ 2 อย่าง = Error Map(1 -> John, 2 -> Ann, 3 -> May)
```

```
println(mymap(2)) // use key to get value Ann
```

```
//println(mymap(0)) // non existing key throws exception Set(1, 2, 3)
```

```
println(mymap.keys) มัน set ของ key Iterable(John, Ann, May)
```

```
println(mymap.values) false
```

```
println(mymap.isEmpty) false
```

```
println(mymap.contains(0)) key = 1, value = John
```

มันจะไล่ดูทั้ง keys

```
mymap.keys.foreach{ key => //iterate
```

key = 2, value = Ann

key = 3, value = May

```
println("key = " + key + ", value = " + mymap(key))
```

```
}
```

```
}
```

```
}
```

```
object MyMap02 {  
  val m1: Map[Int,String] = Map(1 -> "John", 2 -> "Ann", 3 -> "May")  
  val m2 = Map(2 -> "Kim", 4 -> "Lee", 1 -> "Ann", 5 -> "Penguin")  
  
  def main(args: Array[String]): Unit = {  
    println(m1 ++ m2) // concat  
    println(m1.head)  
    println(m1.tail)  
    println(m1.size)  
  }  
}
```

```
HashMap(5 -> Penguin, 1 -> Ann, 2 -> Kim, 3 -> May, 4 -> Le  
(1,John)  
Map(2 -> Ann, 3 -> May)  
3
```

TUPLE

- Collection of values.
- Can contain different data type.
- Tuple is immutable!
- Each tuple can only contain upto 22 data.
- Position in a tuple starts from 1. *๖ เริ่มที่เลข 1 (ไม่ใช่! 0 นะ)*
- Data in a map is actually a tuple.

```
object MyTouple {
  val mytuple = (1,2,"A",3.14,false)
  val mytuple2 = new Tuple4("SS",7.33,"Man",tuple(2,3))
```

==> สิ่งลง 4 ตัว — จำนวนของ "ต้องตรงกัน"

```
def main(args: Array[String]): Unit = {
```

```
  println(mytuple)
```

```
  println(mytuple._3) //data from position 3
```

```
  println(mytuple2._4)
```

```
  println(mytuple2._4._2)
```

```
  println("-----")
```

```
  mytuple.productIterator.foreach{ //iterate
```

```
    value => println(value)
```

```
  }
```

```
  println("-----")
```

```
  println(1 -> "jojo" -> 1897) //nested tuple (map notation)
```

```
}
```

```
}
```

```
(1,2,A,3.14,false)
```

```
A
```

```
(2,3)
```

```
3
```

```
-----
```

```
1
```

```
2
```

```
A
```

```
3.14
```

```
false
```

```
-----
```

```
((1,jojo),1897)
```

OPTION TYPE

- Normally used as a return type
 - For example: return an answer or None

```
object MyOption {  
  val l1 = List(1,2,3)  
  val m1 = Map(1 -> "One", 2 -> "Two")  
  def main(args: Array[String]): Unit = {  
    println(l1.find(_ > 1)) //if there is an answer, Some(2)  
    println(l1.find(_ > 1).get) //return ค่าที่เจอ some → 2  
    println(l1.find(_ > 3)) → None Value  
  
    println(m1.get(1)) → Some(One)  
    println(m1.get(1).get) → One  
    println(m1.get(0)) → None (เพราะไม่มีค่า)  
    println(m1.get(0).getOrElse("No value found")) → No value found  
  }  
}
```

ถ้าเป็น m1.get(0).get จะ Error!

ถ้าเป็น None ให้คืนค่าของเดิม


```

object MyOption2 {
  val l1 = List(1,2,3)
  val opt1: Option[Int] = None
  val opt2: Option[Int] = Some(2)

  def findPos(v:Int, l:List[Int]): Option[Int] = {
    return findPos(v,l, count = 0)
  }

  def findPos(v:Int, l:List[Int], count: Int):Option[Int] = {
    if(l.isEmpty) return None
    if(v == l.head) return Some(count)
    else {
      return findPos(v,l.tail,count+1)
    }
  }

  def main(args: Array[String]): Unit = {
    println(opt1.isEmpty)
    println(opt1.getOrElse("NO"))
    println(findPos(2,l1))
    println(findPos(4,l1))
  }
}

```

true

NO

Some(1)

None

index



```

1 startup_stm32...
2 system_stm32...
3 Console
4 main.h
5
6 01 //
7 02 #define __GNUC__
8 03 #define __cplusplus
9 04 #define __cplusplus
10 05 #include <stdio.h>
11 06 #include <stdlib.h>
12 07 #include <string.h>
13 08 #include <math.h>
14 09 #include <stdint.h>
15 10 #include <stdbool.h>
16 11 #include <unistd.h>
17 12 #include <sys/types.h>
18 13 #include <sys/stat.h>
19 14 #include <fcntl.h>
20 15 #include <pthread.h>
21 16 #include <semaphore.h>
22 17 #include <sys/time.h>
23 18 #include <sys/mman.h>
24 19 #include <sys/ioctl.h>
25 20 #include <sys/uio.h>
26 21 #include <sys/queue.h>
27 22 #include <sys/random.h>
28 23 #include <sys/epoll.h>
29 24 #include <sys/eventfd.h>
30 25 #include <sys/signalfd.h>
31 26 #include <sys/timerfd.h>
32 27 #include <sys/prctl.h>
33 28 #include <sys/procfs.h>
34 29 #include <sys/ptrace.h>
35 30 #include <sys/auxv.h>
36 31 #include <sys/utsname.h>
37 32 #include <sys/xattr.h>
38 33 #include <sys/fs.h>
39 34 #include <sys/mount.h>
40 35 #include <sys/quota.h>
41 36 #include <sys/swap.h>
42 37 #include <sys/vfs.h>
43 38 #include <sys/proc.h>
44 39 #include <sys/procfs.h>
45 40 #include <sys/procfs.h>
46 41 #include <sys/procfs.h>
47 42 #include <sys/procfs.h>
48 43 #include <sys/procfs.h>
49 44 #include <sys/procfs.h>
50 45 #include <sys/procfs.h>
51 46 #include <sys/procfs.h>
52 47 #include <sys/procfs.h>
53 48 #include <sys/procfs.h>
54 49 #include <sys/procfs.h>
55 50 #include <sys/procfs.h>
56 51 #include <sys/procfs.h>
57 52 #include <sys/procfs.h>
58 53 #include <sys/procfs.h>
59 54 #include <sys/procfs.h>
60 55 #include <sys/procfs.h>
61 56 #include <sys/procfs.h>
62 57 #include <sys/procfs.h>
63 58 #include <sys/procfs.h>
64 59 #include <sys/procfs.h>
65 60 #include <sys/procfs.h>
66 61 #include <sys/procfs.h>
67 62 #include <sys/procfs.h>
68 63 #include <sys/procfs.h>
69 64 #include <sys/procfs.h>
70 65 #include <sys/procfs.h>
71 66 #include <sys/procfs.h>
72 67 #include <sys/procfs.h>
73 68 #include <sys/procfs.h>
74 69 #include <sys/procfs.h>
75 70 #include <sys/procfs.h>
76 71 #include <sys/procfs.h>
77 72 #include <sys/procfs.h>
78 73 #include <sys/procfs.h>
79 74 #include <sys/procfs.h>
80 75 #include <sys/procfs.h>
81 76 #include <sys/procfs.h>
82 77 #include <sys/procfs.h>
83 78 #include <sys/procfs.h>
84 79 #include <sys/procfs.h>
85 80 #include <sys/procfs.h>
86 81 #include <sys/procfs.h>
87 82 #include <sys/procfs.h>
88 83 #include <sys/procfs.h>
89 84 #include <sys/procfs.h>
90 85 #include <sys/procfs.h>
91 86 #include <sys/procfs.h>
92 87 #include <sys/procfs.h>
93 88 #include <sys/procfs.h>
94 89 #include <sys/procfs.h>
95 90 #include <sys/procfs.h>
96 91 #include <sys/procfs.h>
97 92 #include <sys/procfs.h>
98 93 #include <sys/procfs.h>
99 94 #include <sys/procfs.h>
100 95 #include <sys/procfs.h>
101 96 #include <sys/procfs.h>
102 97 #include <sys/procfs.h>
103 98 #include <sys/procfs.h>
104 99 #include <sys/procfs.h>
105 100 #include <sys/procfs.h>
106 101 #include <sys/procfs.h>
107 102 #include <sys/procfs.h>
108 103 #include <sys/procfs.h>
109 104 #include <sys/procfs.h>
110 105 #include <sys/procfs.h>
111 106 #include <sys/procfs.h>
112 107 #include <sys/procfs.h>
113 108 #include <sys/procfs.h>
114 109 #include <sys/procfs.h>
115 110 #include <sys/procfs.h>
116 111 #include <sys/procfs.h>
117 112 #include <sys/procfs.h>
118 113 #include <sys/procfs.h>
119 114 #include <sys/procfs.h>
120 115 #include <sys/procfs.h>
121 116 #include <sys/procfs.h>
122 117 #include <sys/procfs.h>
123 118 #include <sys/procfs.h>
124 119 #include <sys/procfs.h>
125 120 #include <sys/procfs.h>
126 121 #include <sys/procfs.h>
127 122 #include <sys/procfs.h>
128 123 #include <sys/procfs.h>
129 124 #include <sys/procfs.h>
130 125 #include <sys/procfs.h>
131 126 #include <sys/procfs.h>
132 127 #include <sys/procfs.h>
133 128 #include <sys/procfs.h>
134 129 #include <sys/procfs.h>
135 130 #include <sys/procfs.h>
136 131 #include <sys/procfs.h>
137 132 #include <sys/procfs.h>
138 133 #include <sys/procfs.h>
139 134 #include <sys/procfs.h>
140 135 #include <sys/procfs.h>
141 136 #include <sys/procfs.h>
142 137 #include <sys/procfs.h>
143 138 #include <sys/procfs.h>
144 139 #include <sys/procfs.h>
145 140 #include <sys/procfs.h>
146 141 #include <sys/procfs.h>
147 142 #include <sys/procfs.h>
148 143 #include <sys/procfs.h>
149 144 #include <sys/procfs.h>
150 145 #include <sys/procfs.h>
151 146 #include <sys/procfs.h>
152 147 #include <sys/procfs.h>
153 148 #include <sys/procfs.h>
154 149 #include <sys/procfs.h>
155 150 #include <sys/procfs.h>
156 151 #include <sys/procfs.h>
157 152 #include <sys/procfs.h>
158 153 #include <sys/procfs.h>
159 154 #include <sys/procfs.h>
160 155 #include <sys/procfs.h>
161 156 #include <sys/procfs.h>
162 157 #include <sys/procfs.h>
163 158 #include <sys/procfs.h>
164 159 #include <sys/procfs.h>
165 160 #include <sys/procfs.h>
166 161 #include <sys/procfs.h>
167 162 #include <sys/procfs.h>
168 163 #include <sys/procfs.h>
169 164 #include <sys/procfs.h>
170 165 #include <sys/procfs.h>
171 166 #include <sys/procfs.h>
172 167 #include <sys/procfs.h>
173 168 #include <sys/procfs.h>
174 169 #include <sys/procfs.h>
175 170 #include <sys/procfs.h>
176 171 #include <sys/procfs.h>
177 172 #include <sys/procfs.h>
178 173 #include <sys/procfs.h>
179 174 #include <sys/procfs.h>
180 175 #include <sys/procfs.h>
181 176 #include <sys/procfs.h>
182 177 #include <sys/procfs.h>
183 178 #include <sys/procfs.h>
184 179 #include <sys/procfs.h>
185 180 #include <sys/procfs.h>
186 181 #include <sys/procfs.h>
187 182 #include <sys/procfs.h>
188 183 #include <sys/procfs.h>
189 184 #include <sys/procfs.h>
190 185 #include <sys/procfs.h>
191 186 #include <sys/procfs.h>
192 187 #include <sys/procfs.h>
193 188 #include <sys/procfs.h>
194 189 #include <sys/procfs.h>
195 190 #include <sys/procfs.h>
196 191 #include <sys/procfs.h>
197 192 #include <sys/procfs.h>
198 193 #include <sys/procfs.h>
199 194 #include <sys/procfs.h>
200 195 #include <sys/procfs.h>
201 196 #include <sys/procfs.h>
202 197 #include <sys/procfs.h>
203 198 #include <sys/procfs.h>
204 199 #include <sys/procfs.h>
205 200 #include <sys/procfs.h>
206 201 #include <sys/procfs.h>
207 202 #include <sys/procfs.h>
208 203 #include <sys/procfs.h>
209 204 #include <sys/procfs.h>
210 205 #include <sys/procfs.h>
211 206 #include <sys/procfs.h>
212 207 #include <sys/procfs.h>
213 208 #include <sys/procfs.h>
214 209 #include <sys/procfs.h>
215 210 #include <sys/procfs.h>
216 211 #include <sys/procfs.h>
217 212 #include <sys/procfs.h>
218 213 #include <sys/procfs.h>
219 214 #include <sys/procfs.h>
220 215 #include <sys/procfs.h>
221 216 #include <sys/procfs.h>
222 217 #include <sys/procfs.h>
223 218 #include &
```