

CH16 練習

鄭安翔

ansel_cheng@hotmail.com

練習一 Lambda Expression

1. 開啟 LambdaBasic 專案
2. 檢視下列程式
 - ❑ `com.example.AnalyzerTool` 類別
 - ❑ `com.example.StringAnalyzer` 介面
 - ❑ `com.example.LambdaTest` 主類別
3. 使用 Lambda Expression 撰寫下列比對功能
 - ❑ 包含 'to' 的字串
 - ❑ 'to' 為開頭的字串
 - ❑ 相等於 'to' 的字串
 - ❑ 'to' 為結尾的字串
 - ❑ 包含 'to' 且少於5個字元的字串
 - ❑ 包含 'to' 且多於5個字元的字串

AnalyzerTool / StringAnalyzer

```
AnalyzerTool.java X
1 package com.example;
2
3 public class AnalyzerTool {
4     public void showResult(String[] strList, String searchStr, StringAnalyzer analyzer) {
5         for (String current : strList) {
6             if (analyzer.analyze(current, searchStr)) {
7                 System.out.println("Match: " + current);
8             }
9         }
10    }
11 }
12
```

```
StringAnalyzer.java X
1 package com.example;
2
3 public interface StringAnalyzer {
4
5     public boolean analyze(String target, String searchStr);
6
7 }
8
```

LambdaTest 類別

```
LambdaTest.java X
1 package com.example;
2
3 public class LambdaTest {
4
5     public static void main(String[] args) {
6         String[] strList01 = {"tomorrow", "toto", "to", "timbukto", "the", "hello", "heat"};
7
8         AnalyzerTool stringTool = new AnalyzerTool();
9         String searchStr = "to";
10
11         System.out.println("Searching for: " + searchStr);
12
13         System.out.println("==Contains==");
14         // Your code here
15
16         System.out.println("=
17         // Your code here
18
19         System.out.println("=
20         // Your code here
21
22         System.out.println("=
23         // Your code here
24
25         System.out.println("=
26         // Your Code here
27
28         System.out.println("=
29         // Your code here
30     }
31
32 }
```



```
public static void main(String[] args) {
    String[] strList01 = {"tomorrow", "toto", "to", "timbukto", "the", "hello", "heat"};

    AnalyzerTool stringTool = new AnalyzerTool();
    String searchStr = "to";

    System.out.println("Searching for: " + searchStr);

    System.out.println("==Contains==");
    stringTool.showResult(strList01, searchStr, (t, s) -> t.contains(s));

    System.out.println("==Starts With==");
    stringTool.showResult(strList01, searchStr, (t,s)->t.startsWith(s));

    System.out.println("==Equals==");
    stringTool.showResult(strList01, searchStr, (t,s)->t.equals(s));

    System.out.println("==Ends With==");
    stringTool.showResult(strList01, searchStr, (t,s)->t.endsWith(s));

    System.out.println("==Less than 5==");
    stringTool.showResult(strList01, searchStr, (t,s)->t.contains(s)&& t.length()<5);

    System.out.println("==Greater than 5==");
    stringTool.showResult(strList01, searchStr, (t,s)->t.contains(s)&& t.length()>5);
}
```

測試執行

```
Problems Console X Progress
<terminated> LambdaTest [Java Application]
Searching for: to
==Contains==
Match: tomorrow
Match: toto
Match: to
Match: timbukto
==Starts With==
Match: tomorrow
Match: toto
Match: to
==Equals==
Match: to
==Ends With==
Match: toto
Match: to
Match: timbukto
==Less than 5==
Match: toto
Match: to
==Greater than 5==
Match: tomorrow
Match: timbukto
```

練習二 Lambda Expression

1. 開啟 LambdaTest 專案
2. 檢視下列程式
 - `com.example.Person` 類別
 - `com.example.Gender` 列舉
 - `com.example.LambdaCollectionTest` 主類別
3. 使用 Lambda Expression 定義集合操作規則
 - 以 `LastName` 來作升冪排序
 - 以 `Age` 來作降冪排序
 - 移除所有女性成員
 - 移除年齡小於 35 成員

```

5 public class Person {
6     private String firstName;
7     private String lastName;
8     private int age;
9     private Gender gender;
10
11     public Person(String firstName, String lastName,
12                   int age, Gender gender) {
13         this.firstName = firstName;
14         this.lastName = lastName;
15         this.age = age;
16         this.gender = gender;
17     }
18
19     public String getFirstName() {
20         return firstName;
21     }
22
23     public String getLastName() {
24         return lastName;
25     }
26
27     public Gender getGender() {
28         return gender;
29     }
30
31     public int getAge(){
32         return age;
33     }
34

```

```

1 package com.example;
2
3 public enum Gender {
4     MALE, FEMALE
5 }
6

```

```

35 @Override
36 public String toString(){
37     return "Name: " + firstName + " " + lastName + "\n"
38           + "Age: " + age + " Gender: " + gender + "\n";
39 }
40
41 public static List<Person> createList() {
42     List<Person> people = new ArrayList<>();
43     people.add(new Person("Sean", "Cheng", 43, Gender.MALE));
44     people.add(new Person("Bob", "Baker", 21, Gender.MALE));
45     people.add(new Person("Jane", "Doe", 35, Gender.FEMALE));
46     people.add(new Person("John", "Doe", 27, Gender.MALE));
47     people.add(new Person("James", "Johnson", 45, Gender.MALE));
48     people.add(new Person("Betty", "Jones", 33, Gender.FEMALE));
49     people.add(new Person("Ivy", "Wang", 23, Gender.FEMALE));
50     people.add(new Person("Nicole", "Wei", 42, Gender.FEMALE));
51     return people;
52 }
53
54 }

```

LambdaCollectionTest 類別

```
LambdaCollectionTest.java X
1 package com.example;
2
3 import java.util.*;
4
5 public class LambdaCollectionTest {
6
7     public static void main(String[] args) {
8         List<Person> personList = Person.createList();
9
10        // 使用 Lambda Expression 定義以LastName來升冪排序
11
12        // 使用 Lambda Expression 定義以Age來降冪排序
13
14        // 使用 Lambda Expression 定義移除所有女性成員
15
16        // 使用 Lambda Expression 定義移除年齡小於35成員
17
18    }
19
20 }
```


LambdaCollectionTest 類別

LambdaCollectionTest.java X

```
5 public class LambdaCollectionTest {
6
7     public static void main(String[] args) {
8         List<Person> personList = Person.createList();
9
10        // 使用 Lambda Expression 定義以LastName來升冪排序
11        System.out.println("==== LastName 升冪排序 ====");
12        Collections.sort(personList, (p1,p2)->p1.getLastName().compareTo(p2.getLastName()));
13
14        for(Person p : personList) {
15            System.out.println(p);
16        }
17
18        // 使用 Lambda Expression 定義以Age來降冪排序
19        System.out.println("==== Age 降冪排序 ====");
20        Collections.sort(personList, (p1,p2)->p2.getAge()-p1.getAge());
21
22        for(Person p : personList) {
23            System.out.println(p);
24        }
25    }
```

LambdaCollectionTest 類別

```
26      // 使用 Lambda Expression 定義移除所有女性成員
27      System.out.println("==== 男性成員 ====");
28      personList.removeIf(p->p.getGender()==Gender.FEMALE);
29
30      for(Person p : personList) {
31          System.out.println(p);
32      }
33
34      // 使用 Lambda Expression 定義移除年齡小於35成員
35      System.out.println("==== 年齡大於等於35歲成員 ====");
36      personList = Person.createList();
37      personList.removeIf(p->p.getAge()<35);
38
39      for(Person p : personList) {
40          System.out.println(p);
41      }
42
43  }
44
45 }
```

測試執行

Problems Console X Progress Git Repos
<terminated> LambdaCollectionTest [Java Application] C:\

==== LastName 升冪排序 ====

Name: Bob Baker
Age: 21 Gender: MALE

Name: Sean Cheng
Age: 43 Gender: MALE

Name: Jane Doe
Age: 35 Gender: FEMALE

Name: John Doe
Age: 27 Gender: MALE

Name: James Johnson
Age: 45 Gender: MALE

Name: Betty Jones
Age: 33 Gender: FEMALE

Name: Ivy Wang
Age: 23 Gender: FEMALE

Name: Nicole Wei
Age: 42 Gender: FEMALE

==== Age 降冪排序 ====

Name: James Johnson
Age: 45 Gender: MALE

Name: Sean Cheng
Age: 43 Gender: MALE

Name: Nicole Wei
Age: 42 Gender: FEMALE

Name: Jane Doe
Age: 35 Gender: FEMALE

Name: Betty Jones
Age: 33 Gender: FEMALE

Name: John Doe
Age: 27 Gender: MALE

Name: Ivy Wang
Age: 23 Gender: FEMALE

Name: Bob Baker
Age: 21 Gender: MALE

==== 男性成員 ====

Name: James Johnson
Age: 45 Gender: MALE

Name: Sean Cheng
Age: 43 Gender: MALE

Name: John Doe
Age: 27 Gender: MALE

Name: Bob Baker
Age: 21 Gender: MALE

==== 年齡大於等於35歲成員 ====

Name: Sean Cheng
Age: 43 Gender: MALE

Name: Jane Doe
Age: 35 Gender: FEMALE

Name: James Johnson
Age: 45 Gender: MALE

Name: Nicole Wei
Age: 42 Gender: FEMALE

練習三 Lambda實作內建函式介面

1. 開啟 LambdaTest 專案
2. 檢視下列程式
 - `com.example.Person` 類別
 - `com.example.Gender` 列舉
 - `com.example.LambdaBuildInTest` 主類別
3. 修改 `LambdaBuildInTest` 類別
 - 使用使用Lambda Expression 定義下列函式介面
 - 定義 `Function` 函式介面, 傳回成員稱謂
 - 姓名前加上Ms./Mr.
 - 定義 `Predicate` 函式介面, 篩選30歲以下成員
 - 定義 `Consumer` 函式介面, 以自訂格式列印成員
 - `FirstName(age)`
4. 測試、執行

Person.java X

```
5 public class Person {
6     private String firstName;
7     private String lastName;
8     private int age;
9     private Gender gender;
10
11     public Person(String firstName, String lastName,
12                   int age, Gender gender) {
13         this.firstName = firstName;
14         this.lastName = lastName;
15         this.age = age;
16         this.gender = gender;
17     }
18
19     public String getFirstName() {
20         return firstName;
21     }
22
23     public String getLastName() {
24         return lastName;
25     }
26
27     public Gender getGender() {
28         return gender;
29     }
30
31     public int getAge(){
32         return age;
33     }
34
```

Gender.java X

```
1 package com.example;
2
3 public enum Gender {
4     MALE, FEMALE
5 }
6
```

```
35 @Override
36 public String toString(){
37     return "Name: " + firstName + " " + lastName + "\n"
38           + "Age: " + age + " Gender: " + gender + "\n";
39 }
40
41 public static List<Person> createList() {
42     List<Person> people = new ArrayList<>();
43     people.add(new Person("Sean", "Cheng", 43, Gender.MALE));
44     people.add(new Person("Bob", "Baker", 21, Gender.MALE));
45     people.add(new Person("Jane", "Doe", 35, Gender.FEMALE));
46     people.add(new Person("John", "Doe", 27, Gender.MALE));
47     people.add(new Person("James", "Johnson", 45, Gender.MALE));
48     people.add(new Person("Betty", "Jones", 33, Gender.FEMALE));
49     people.add(new Person("Ivy", "Wang", 23, Gender.FEMALE));
50     people.add(new Person("Nicole", "Wei", 42, Gender.FEMALE));
51     return people;
52 }
53
54 }
```

LambdaBuildInTest 類別

```
LambdaBuildInTest.java ×
1 package com.example;
2
3 import java.util.List;
4 import java.util.function.*;
5
6 public class LambdaBuildInTest {
7
8     public static void main(String[] args) {
9         List<Person> personList = Person.createList();
10
11         // 使用 Lambda Expression 定義以Function 函式介面,傳回稱謂(姓名前加上Ms./Mr.)
12
13
14
15
16         // 使用 Lambda Expression 定義Predicate 函式介面,篩選列印30歲以下的Person資訊
17
18
19
20
21         // 使用 Lambda Expression 定義以Consumer 函式介面以FirstName(age)格式來列印Person資訊
22
23
24
25
26     }
27
28 }
29
```

LambdaBuildInTest 類別

```
LambdaBuildInTest.java X
1 package com.example;
2
3 import java.util.List;
4 import java.util.function.*;
5
6 public class LambdaBuildInTest {
7
8     public static void main(String[] args) {
9         List<Person> personList = Person.createList();
10
11         // 使用 Lambda Expression 定義以Function 函式介面,傳回稱謂(姓名前加上Ms./Mr.)
12         System.out.println("==== 成員稱謂 ====");
13         Function<Person, String> nameFormat = p ->
14             (p.getGender()==Gender.MALE?"Mr. ":"Ms. ") + p.getLastName();
15         for(Person p:personList)
16             System.out.println(nameFormat.apply(p));
17     }
18 }
```

LambdaBuildInTest 類別

```
18 // 使用 Lambda Expression 定義Predicate 函式介面,篩選列印30歲以下的Person資訊
19 System.out.println("==== 年齡小於30歲成員 ====");
20 Predicate<Person> youth = p ->p.getAge()<30;
21 for(Person p:personList)
22     if(youth.test(p))
23         System.out.println(p);
24
25 // 使用 Lambda Expression 定義以Consumer 函式介面以FirstName(age)格式來列印Person資訊
26 System.out.println("==== 自訂列印格式 ====");
27 Consumer<Person> myPrint = p ->
28     System.out.printf("%s(%d)%n", p.getFirstName(), p.getAge());
29 for(Person p:personList)
30     myPrint.accept(p);
31
32 }
33
34 }
```


測試執行

```
Problems Console X Progress
<terminated> LambdaBuildInTest [Java Applet]

==== 成員稱謂 ====
Mr. Cheng
Mr. Baker
Ms. Doe
Mr. Doe
Mr. Johnson
Ms. Jones
Ms. Wang
Ms. Wei
```

```
==== 年齡小於30歲成員 ====
Name: Bob Baker
Age: 21  Gender: MALE

Name: John Doe
Age: 27  Gender: MALE

Name: Ivy Wang
Age: 23  Gender: FEMALE
```

```
==== 自訂列印格式 ====
Sean(43)
Bob(21)
Jane(35)
John(27)
James(45)
Betty(33)
Ivy(23)
Nicole(42)
```