

6. 고객관리 프로그램 1

이 장에서는 지금까지 배운 내용을 이용하여 고객관리 애플리케이션을 만들어 봅니다.

6장의 주요 내용입니다.

- 요구사항 정의
- 변수 선언
- 메뉴 선택

6.1. 고객관리 프로그램

지금 까지 배운 내용을 토대로 고객의 정보를 관리하는 프로그램을 만들 수 있습니다. 아직 파일 입출력을 배우지 않았기 때문에 파일에 데이터를 저장할 수는 없을 것입니다. 지금까지 배운 내용으로 고객의 정보를 관리하는 프로그램을 만들어야 하므로 고객의 정보를 저장하는 자료구조는 배열을 이용하겠습니다.

그리고 지금은 객체지향의 개념을 아직 배우기 전 이기 때문에 고객 정보를 저장하는 배열들을 정의하여야 합니다.

6.1.1. 요구사항 정의

고객의 정보는 이름, 성별, 이메일, 출생년도가 있습니다. 고객의 정보를 입력받아 배열에 저장해야 합니다. 이름은 문자열로 저장하며, 성별은 남자는 M, 여자는 F로 저장합니다. 이메일은 문자열로 저장하며 태어난 출생년도는 정수로 저장합니다.

고객 관리 프로그램은 고객의 정보를 저장, 조회, 수정, 삭제할 수 있는 기능이 있어야 합니다. 고객 정보를 파일에 저장하는 기능을 구현하지 않아도 됩니다.

I를 눌러 고객의 정보를 입력받도록 하며, 저장된 고객 정보는 P 또는 N을 눌러 이전 고객 정보 또는 다음 고객정보를 조회할 수 있어야 합니다. 조회한 고객 정보는 U를 눌러 새로운 정보로 수정할 수 있어야 합니다. D를 누르면 조회한 고객 정보를 배열에서 삭제해야 합니다. 프로그램의 종료는 Q를 누릅니다.

6.1.2. 변수 선언 및 메뉴 선택

먼저 고객의 정보를 저장할 변수를 선언해야 합니다. 아직 우리는 객체와 클래스에 대해 배우지 않았으므로 고객의 정보를 저장하기 위해 클래스를 사용할 수 없습니다. 고객의 정보 이름, 성별, 이메일, 출생년도를 저장하기 위해 배열을 이용했습니다. 메뉴를 입력받기 위해 Scanner 클래스를 이용했습니다. 그리고 메뉴를 선택했을 경우 실행할 코드는 switch문을 이용해서 구현했습니다.

CustomerManager.java

```
1: import java.util.Scanner;
2:
3: public class CustomerManager {
4:     //배열에 저장할 수 있는 최대 고객의 수
5:     static final int MAX = 100;
6:     //고객정보를 저장할 변수를 배열로 선언
7:     static String[] nameList = new String[MAX];    //이름 저장
```

```

8:      static char[] genderList = new char[MAX];          //성별 저장
9:      static String[] emailList = new String[MAX];       //이메일 저장
10:     static int[] birthYearList = new int[MAX];         //출생년도 저장
11:
12:     //배열은 인덱스를 필요로 함
13:     static int index = -1; //배열은 0부터 시작하므로 최초 인덱스는 -1이어야 함
14:
15:     //배열은 처음 선언한 크기보다 같거나 적은 개수의 자료를 저장
16:     //그래서 현재 데이터가 몇 개 저장되어있는 지 알 수 있는 변수 선언
17:     static int count = 0; //개수
18:
19:     //기본 입력장치로부터 데이터를 입력받기 위해 Scanner 객체 생성
20:     static Scanner scan = new Scanner(System.in);
21:
22:     public static void main(String[] args) {
23:
24:         while(true) {
25:
26:             System.out.printf("\n[INFO] 고객 수 : %d, 인덱스 : %d\n", count, index);
27:             System.out.println("메뉴를 입력하세요.");
28:             System.out.println("(I)nsert, (P)revious, (N)ext, " +
29:                 "(C)urrent, (U)pdate, (D)elete, (Q)uit");
30:             System.out.print("메뉴 입력: ");
31:             String menu = scan.next();
32:             menu = menu.toLowerCase();          //입력한 문자열을 모두소문자로 변환
33:             switch(menu.charAt(0)) {
34:             case 'i':
35:                 System.out.println("고객정보 입력을 시작합니다.");
36:
37:                 break;
38:             case 'p' :
39:                 System.out.println("이전 데이터를 출력합니다.");
40:
41:                 break;
42:             case 'n' :
43:                 System.out.println("다음 데이터를 출력합니다.");
44:
45:                 break;
46:             case 'c' :
47:                 System.out.println("현재 데이터를 출력합니다.");
48:
49:                 break;
50:             case 'u' :
51:                 System.out.println("데이터를 수정합니다.");
52:
53:                 break;
54:             case 'd' :
55:                 System.out.println("데이터를 삭제합니다.");
56:
57:                 break;

```

```

58:         case 'q' :
59:             System.out.println("프로그램을 종료합니다.");
60:             scan.close();           //Scanner 객체를 닫아준다.
61:             System.exit(0);         //프로그램을 종료시킨다.
62:             break;
63:         default :
64:             System.out.println("메뉴를 잘 못 입력했습니다.");
65:         } //end switch
66:     } //end while
67: } //end main
68:
69: }

```

6.1.3. 완성된 코드

이 예제는 고객의 정보를 배열에 저장하기 때문에 프로그램을 종료한 다음 다시 실행시키면 이전에 저장했던 정보는 사라집니다.

다음은 완성된 코드입니다.

CustomerManager.java

```

1: import java.util.Scanner;
2:
3: public class CustomerManager {
4:
5:     //배열에 저장할 수 있는 최대 고객의 수
6:     static final int MAX = 100;
7:     //고객정보를 저장할 변수를 배열로 선언
8:     static String[] nameList = new String[MAX];           //이름 저장
9:     static char[] genderList = new char[MAX];             //성별 저장
10:    static String[] emailList = new String[MAX];           //이메일 저장
11:    static int[] birthYearList = new int[MAX];             //출생년도 저장
12:
13:    //배열은 인덱스를 필요로 함
14:    static int index = -1; //배열은 0부터 시작하므로 최초 인덱스는 -1이어야 함
15:
16:    //배열은 처음 선언한 크기보다 같거나 적은 개수의 자료를 저장
17:    //그래서 현재 데이터가 몇 개 저장되어있는 지 알 수 있는 변수 선언
18:    static int count = 0; //개수
19:
20:    //기본 입력장치로부터 데이터를 입력받기 위해 Scanner 객체 생성
21:    static Scanner scan = new Scanner(System.in);
22:
23:    public static void main(String[] args) {
24:
25:        while(true) {
26:

```

```

27:      System.out.printf("\n[INFO] 고객 수 : %d, 인덱스 : %d\n", count, index);
28:      System.out.println("메뉴를 입력하세요.");
29:      System.out.println("(I)nsert, (P)revious, (N)ext, " +
30:          "(C)urrent, (U)pdate, (D)elete, (Q)uit");
31:      System.out.print("메뉴 입력: ");
32:      String menu = scan.next();
33:      menu = menu.toLowerCase();          //입력한 문자열을 모두소문자로 변환
34:      switch(menu.charAt(0)) {
35:      case 'i':
36:          System.out.println("고객정보 입력을 시작합니다.");
37:          insertCustomerData();
38:          System.out.println("고객정보를 입력했습니다.");
39:          break;
40:      case 'p' :
41:          System.out.println("이전 데이터를 출력합니다.");
42:          if(index <= 0) {
43:              System.out.println("이전 데이터가 존재하지 않습니다.");
44:          }else {
45:              index--;
46:              printCustomerInfo(index);
47:          }
48:          break;
49:      case 'n' :
50:          System.out.println("다음 데이터를 출력합니다.");
51:          if(index >= count-1) {
52:              System.out.println("다음 데이터가 존재하지 않습니다.");
53:          }else {
54:              index++;
55:              printCustomerInfo(index);
56:          }
57:          break;
58:      case 'c' :
59:          System.out.println("현재 데이터를 출력합니다.");
60:          if( (index >= 0) && (index < count)) {
61:              printCustomerInfo(index);
62:          }else {
63:              System.out.println("출력할 데이터가 선택되지 않았습니다.");
64:          }
65:          break;
66:      case 'u' :
67:          System.out.println("데이터를 수정합니다.");
68:          if( (index >= 0) && (index < count)) {
69:              System.out.println(index + "번째 데이터를 수정합니다.");
70:              updateCustomerData(index);
71:          }else {
72:              System.out.println("수정할 데이터가 선택되지 않았습니다.");
73:          }
74:          break;
75:      case 'd' :
76:          System.out.println("데이터를 삭제합니다.");

```

```

77:         if( (index >= 0) && (index < count)) {
78:             System.out.println(index + "번째 데이터를 삭제합니다.");
79:             deleteCustomerData(index);
80:         }else {
81:             System.out.println("삭제할 데이터가 선택되지 않았습니다.");
82:         }
83:         break;
84:     case 'q' :
85:         System.out.println("프로그램을 종료합니다.");
86:         scan.close();           //Scanner 객체를 닫아준다.
87:         System.exit(0);         //프로그램을 종료시킨다.
88:         break;
89:     default :
90:         System.out.println("메뉴를 잘 못 입력했습니다.");
91:     } //end switch
92: } //end while
93: } //end main
94:
95: public static void insertCustomerData() {
96:     System.out.print("이름 : ");
97:     String name = scan.next();
98:     System.out.print("성별(M/F) : ");
99:     char gender = scan.next().charAt(0);
100:    System.out.print("이메일 : ");
101:    String email = scan.next();
102:    System.out.print("출생년도 : ");
103:    int birthYear = scan.nextInt();
104:
105:    //고객 객체를 배열에 저장
106:    //count 번째 배열에 객체 저장 후 count 값을 증가시켜야 함
107:    nameList[count] = name;
108:    genderList[count] = gender;
109:    emailList[count] = email;
110:    birthYearList[count] = birthYear;
111:    count++;
112: }
113:
114: //고객데이터 출력
115: public static void printCustomerInfo(int index) {
116:     System.out.println("=====CUSTOMER INFO=====");
117:     System.out.println("이름 : " + nameList[index]);
118:     System.out.println("성별 : " + genderList[index]);
119:     System.out.println("이메일 : " + emailList[index]);
120:     System.out.println("출생년도 : " + birthYearList[index]);
121:     System.out.println("=====");
122: }
123:
124: //index 위치의 고객정보를 삭제합니다.
125: public static void deleteCustomerData(int index) {
126:     for(int i=index; i<count-1; i++) {

```

```
127:     nameList[i] = nameList[i+1];
128:     genderList[i] = genderList[i+1];
129:     emailList[i] = emailList[i+1];
130:     birthYearList[i] = birthYearList[i+1];
131: }
132:     count--;
133: }
134:
135: //index 위치의 고객 정보를 수정합니다.
136: public static void updateCustomerData(int index) {
137:     System.out.println("-----UPDATE CUSTOMER INFO-----");
138:     System.out.print("이름(" + nameList[index] + ") :");
139:     nameList[index] = scan.next();
140:
141:     System.out.print("성별(" + genderList[index] + ") :");
142:     genderList[index] = scan.next().charAt(0);
143:
144:     System.out.print("이메일(" + emailList[index] + ") :");
145:     emailList[index] = scan.next();
146:
147:     System.out.print("출생년도(" + birthYearList[index] + ") :");
148:     birthYearList[index] = scan.nextInt();
149: }
150:
151: } //end class
```

Scanner

JDK 1.5부터 사용 가능한 클래스입니다. Scanner 클래스를 이용하면 시스템으로부터 데이터를 쉽게 입력 받을 수 있습니다. 뿐만 아니라 Scanner 클래스는 nextInt(), nextDouble(), nextBoolean() 등 기본 데이터형에 맞는 메서드들이 정의되어 있습니다. Scanner 클래스가 추가되기 전과 후의 코드를 비교해 보겠습니다.

- JDK 1.4 까지

```
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
String input = br.readLine();
int num = Integer.parseInt(br.readLine());
```

- JDK 1.5 이후

```
Scanner s = new Scanner(System.in);
String input = s.nextLine();
int num = s.nextInt();
```

6.2. 마인드맵 정리

