■ STEP_15/H1/H1_ByteIO.java : 바이트 단위 파일 IO

FileOutputStream 클래스

FileInputStream 클래스

H1_ByteIO(메인) 클래스

- "data.bin" 파일에 대한 FileOutputStream 클래스의 객체를 생성한다(fos).
- fos의 write() 메소드를 이용하여 다음 데이터를 "data.bin" 파일에 출력(저장)한다.
  . 0x01, 0x02, 0x03, 0x80, 0x81, 0x82
- fos의 close() 메소드를 이용하여 파일을 닫는다.

- "data.bin" 파일에 대한 FileInputStream 클래스의 객체를 생성한다(fin).
- fin의 read() 메소드를 이용하여 6 바이트를 차례로 읽어 화면에 출력한다.
  . 0x01, 0x02, 0x03, 0x80, 0x81, 0x82
- fin의 close() 메소드를 이용하여 파일을 닫는다.

// STEP_15/H1/H1_ByteIO.java

■ STEP_15/H2/H2_DataIO.java : 기본 타입별 데이터 파일 IO

DataOutputStream 클래스 ← FileOutputStream 클래스

DataInputStream ← FileInputStream 클래스

H2_DataIO(메인) 클래스

- "data.bin" 파일에 대한 DataOutputStream 클래스의 객체를 생성한다(dos).
- dos의 write???() 메소드를 이용하여 다음 데이터를 "data.bin" 파일에 출력(저장)한다.
  . true(boolean), 65(byte), 0x42(short), C'(char), 0104(int),
    69L(long), 1e-2f(float), 0.02(double)
- dos의 close() 메소드를 이용하여 파일을 닫는다.

- "data.bin" 파일에 대한 DataInputStream 클래스의 객체를 생성한다(din).
- din의 read???() 메소드를 이용하여 위에서 출력한 데이터 8개를 차례로 읽어 화면에 출력한다.
- din의 close() 메소드를 이용하여 파일을 닫는다.

// STEP_15/H2/H2_DataIO.java

■ STEP_15/H3/H3_CharIO.java : 문자 단위 파일 IO

FileWriter 클래스

FileReader 클래스

H3_CharIO(메인) 클래스

- “data.txt” 파일에 대한 FileWriter 클래스의 객체를 생성한다(fw).
- fw의 write() 메소드를 이용하여 다음 문자들을 “data.txt” 파일에 출력(저장)한다.
  . ‘A’, ‘B’, ‘C’, ‘X’, ‘Y’, ‘Z’
- fw의 close() 메소드를 이용하여 파일을 닫는다.

- “data.txt” 파일에 대한 FileReader 클래스의 객체를 생성한다(fr).
- fr의 read() 메소드를 이용하여 위에서 출력한 문자 6개를 차례로 읽어 화면에 출력한다.
- fr의 close() 메소드를 이용하여 파일을 닫는다.

// STEP_15/H3/H3_CharIO.java

■ STEP_15/H4/H4_StringIO.java : 문자열 단위 파일 IO

BufferedWriter 클래스 ← FileWriter 클래스

BufferedReader 클래스 ← FileReader 클래스

H4_StringIO(메인) 클래스

- "data.txt" 파일에 대한 BufferedWriter 클래스의 객체를 생성한다(bw).
- bw의 write() 메소드를 이용하여 다음 문자열들을 "data.txt" 파일에 출력(저장)한다.
  . "ABC↵", "XYZ↵", "I am happy.↵"
- bw의 close() 메소드를 이용하여 파일을 닫는다.

- "data.txt" 파일에 대한 BufferedReader 클래스의 객체를 생성한다(br).
- br의 readLine() 메소드를 이용하여 위에서 출력한 문자열 두 개를 차례로 읽어 화면에 출력한다.
- fb의 close() 메소드를 이용하여 파일을 닫는다.

// STEP_15/H4/H4_StringIO.java

■ STEP_15/H5/H5_Token.java : 토큰(단어) 분리

BufferedReader 클래스 ←
FileReader 클래스

StringTokenizer 클래스 ←
line

H5_Token(메인) 클래스

- 메모장을 이용하여 “data.txt” 파일에 아래의 문자열들을 편집 저장한다.
- “I,am happy.↵”, “You are,happy.↵”, “He is\tunhappy.↵”

- “data.txt” 파일에 대한 bufferedReader 클래스의 객체를 생성한다(br).
- br의 readLine() 메소드를 이용하여 위에서 출력한 문자열을 하나씩 읽어가면서(line)
  . line에 대하여 StringTokenizer 객체를 생성한다(tkn). ← 구분(분리)자 “ ,\t\n\r”
  . tkn의 nextToken() 메소드로 단어(token)들을 분리하여 화면에 출력한다.
- br의 close() 메소드를 이용하여 파일을 닫는다.

// STEP_15/H5/H5_Token.java

■ STEP_15/H6/H6_Scanner.java : 스캐너 입력

Scanner 클래스 ←
File 클래스

H6_Scanner(메인) 클래스

- 메모장을 이용하여 "data.txt" 파일에 아래의 숫자열들을 편집 저장한다.
- "123 456 789↵ ", "321, 654, 987"↵ "

- "data.txt" 파일에 대하여 Scanner 클래스의 객체를 생성한다(scan).
- scan의 hasNextInt() 및 nextInt() 메소드를 이용하여 정수를 하나씩 읽어(n) 합산한다(sum).
- sum을 화면에 출력한다.
- scan의 close() 메소드를 이용하여 파일을 닫는다.

위의 "data.txt" 파일 대신 System.in 객체를 이용하고, 키보드에서 동일한 숫자열을 입력한다.

// STEP_15/H6/H6_Scanner.java