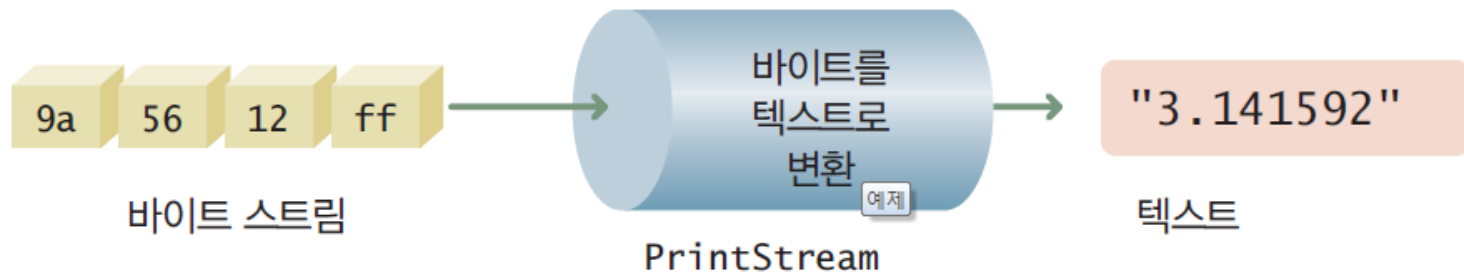


# PrinterStream

경북대학교  
소프트웨어융합과  
배희호 교수  
010-2369-4112  
031-570-9600  
hhbae@kbu.ac.kr

# PrintStream 클래스

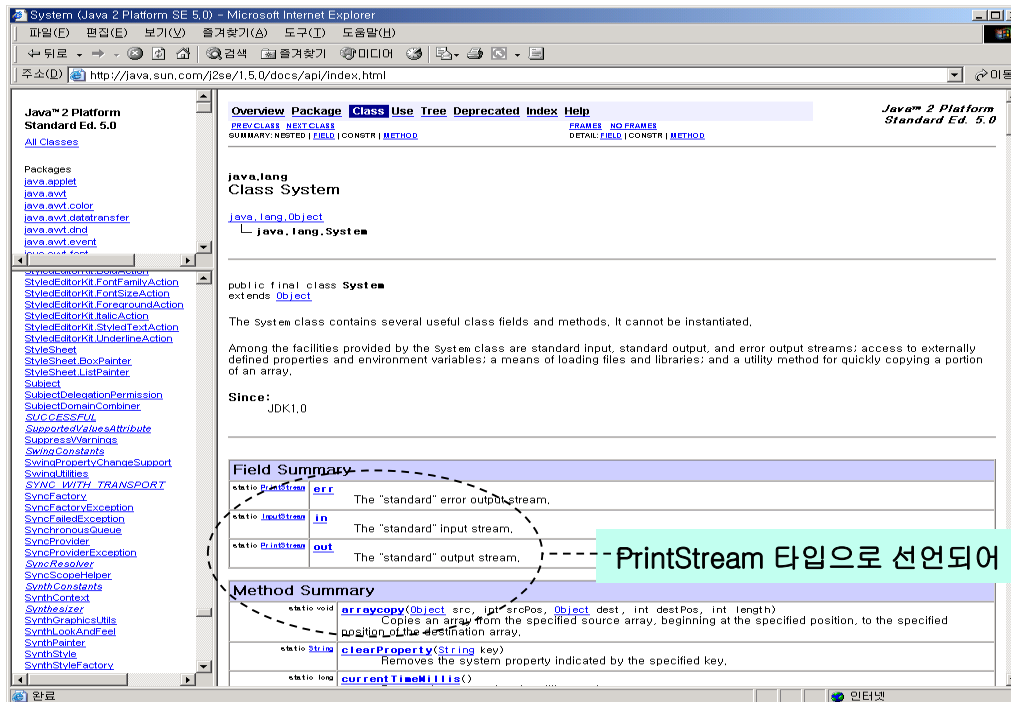
- PrintStream이란 출력장치에 상관없이 조금 더 편하고 쉽게 출력을 도와주는 Stream 클래스임
- PrintStream 클래스는 Filter Stream의 일종으로 OutputStream으로 주어진 출력 Stream에 다양한 출력 기능을 출력하는 기능을 가진 클래스
- 바로 Console창에 출력을 하기 위해서 써왔던 "System.out"이 바로 "PrintStream"



PrintStream의 개념

# PrintStream 클래스

- Data를 Format해서 File로 출력하는 클래스
- 사용 방법은 PrintWriter 클래스와 동일
- JDK 구 버전부터 있던 클래스
- System.out.println(), System.out.print(), System.out.printf() 메소드가 속하는 클래스



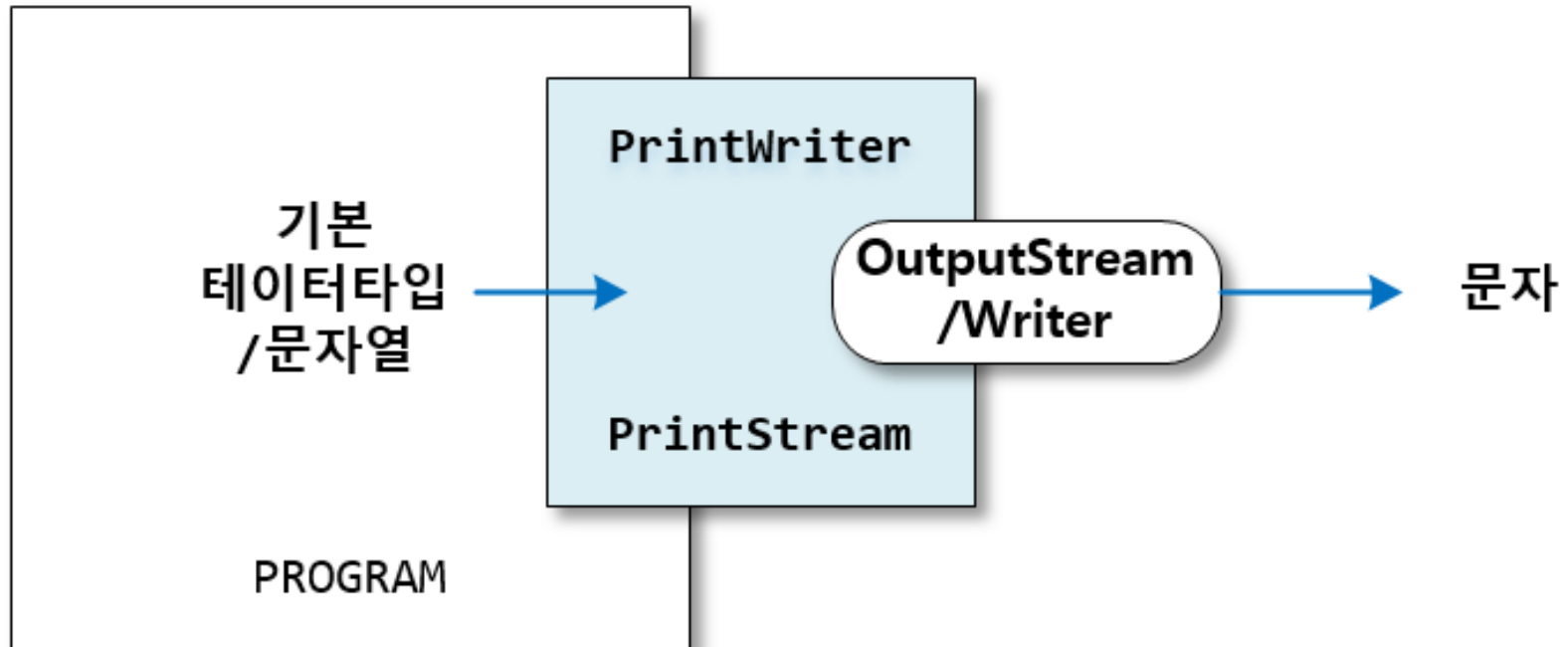
PrintStream 타입으로 선언되어 있는 System 클래스의 out 필드

# PrintStream 클래스

- PrintStream은 모든 자료형을 출력할 수 있는 print(), println() 메소드가 Overloading되어 있음
- Program이 시작되면 장치와 연결된 출력 Stream인 System.out, System.err 객체가 PrintStream 객체임
- JAVA 5.0에서는 PrintStream의 format() 메소드와 printf() 메소드가 추가되어 있기 때문에 System.out.printf()나 System.out.format()을 이용해서 출력문을 작성할 수 있음
- 다른 Stream들과는 다르게 플러쉬(flush) 기능을 자동으로 처리할 수 있는 생성자를 가지고 있음
- 모든 메소드는 예외처리를 하지 않음

# PrintStream 클래스

## ■ PrintStream



# PrintStream 클래스

- 지금까지 많이 써왔던 "print()", "println()" 메소드가 바로 이 "PrintStream"을 쓰는 이유임
- 원래 "Stream" 클래스는 입력과 출력 쌍으로 클래스를 제공
- 하지만 "PrintStream", "PrintWriter" 클래스는 단지 출력 클래스만 존재하고, 대응하는 입력 클래스는 존재하지 않음
- 그런데 그 만큼 사용자 입장에서 만든 것이기 때문에 정교하게 하는 것은 약간 무리

# PrintStream 클래스

## ■ 생성자

생성자	내용
<code>PrintStream(OutputStream out)</code>	주어진 바이트 출력 스트림에 대한 <code>PrintStream</code> 객체를 생성
<code>PrintStream(OutputStream out, boolean autoFlush)</code>	주어진 바이트 출력 스트림에 대해 주어진 자동 flush 기능을 갖는 <code>PrintStream</code> 객체를 생성

# PrintStream 클래스

- System.out.println()의 비밀
  - 지금까지 다만 Text의 출력을 담당한다고만 알고 있던 System.out.println()의 **System.out은 PrintStream 객체임**
  - JAVA에서는 표준 출력을 객체화하기 위해 PrintStream 객체를 생성한 후, System 클래스안에 out이라는 레퍼런스 변수로 가리키고 있었던 것
  - 오류 출력에 사용되는 System.err 객체도 역시 PrintStream 객체



# PrintStream 예제 1

```
public static void main(String[] args) {  
    String path = ".\\data\\test.txt";  
  
    try {  
        PrintStream printStream = new PrintStream(System.out);  
        printStream.println("Test PrintWriter Line 1 ");  
        printStream.printf("%s\\n", "경북대학교 화이팅!");  
        printStream.print("Test PrintWriter Line 3");  
        printStream.close();  
    } catch (Exception e) {  
        System.err.println(e.getMessage());  
    }  
}
```

- ✓ 파일이 없으면 새롭게 만들어 주고, 있으면 기존의 파일 내용을 지워주고 새롭게 생성함
- ✓ System 클래스의 out이라는 static 필드에는 콘솔창에 출력할 수 있는 PrintStream 객체의 참조 값이 들어 있음

# PrintStream 예제 2

```
public static void main(String[] args) {  
    PrintStream printStream = new PrintStream(System.out);  
    printStream.println("    *** 성적표 ***    ");  
    printStream.printf("%3s : %3d %3d %3d %5.1f %n",  
                        "김지영", 92, 87, 95, 91.3f);  
    printStream.printf("%3s : %3d %3d %3d %5.1f %n",  
                        "박현식", 100, 90, 88, 92.7f);  
    printStream.printf("%3s : %3d %3d %3d %5.1f %n",  
                        "최민재", 75, 88, 85, 82.7f);  
    printStream.printf("작성일자 : %1$tY년 %1$tm월 %1$td일",  
                        new GregorianCalendar());  
}
```

# PrintStream 예제 3

## ■ format() 메소드

- format() 메소드는 문자열을 생성하기만 하고, 실제로 출력하지는 않음
- 따라서 출력 결과를 Console에 표시하려면 println() 메소드를 사용하거나, format() 메소드를 호출한 결과를 다른 출력 스트림에 쓸 수 있음

```
public static void main(String[] args) {  
    PrintStream printStream = new PrintStream(System.out);  
    int num = 8;  
    double temp = Math.log(num);  
  
    printStream.format("log(%d)는 %+020.10f 입니다\n", num, temp);  
}
```

log(8)는 +00000002.0794415417 입니다



형식 지정자의 시작

플래그

폭

정밀도

변환

# PrintStream 예제 4

```
public class Main {  
    public static void main(String[] args) {  
        try {  
            PrintStream printStream = new PrintStream("../data//ps.txt");  
            printStream.println(10);  
            printStream.println(3.4);  
            printStream.println('A');  
            printStream.println(true);  
            printStream.println("Hello");  
            printStream.close();  
        } catch (IOException e) {  
            System.out.println(e.getMessage());  
        }  
    }  
}
```

# PrintStream 예제 5

```
public class Main {  
    public static void main(String[] args) {  
        String file = "../data/test.txt";  
        try {  
            PrintStream printStream = new PrintStream(file);  
            printStream.println("변화를 원한다면");  
            printStream.println("제일 먼저 자신이 변화할 수 있다는 것과");  
            printStream.println("변화하기까지 포기하지 않고");  
            printStream.println("계속해서 노력할 수 있다는 것을 믿어야 한다.");  
            printStream.flush();  
            printStream.close();  
            BufferedReader reader = new BufferedReader(new FileReader(file));  
            String line;  
            while ((line = reader.readLine()) != null)  
                System.out.println(line);  
            reader.close();  
        } catch (IOException e) {  
            System.out.println(e.getMessage());  
        }  
    }  
}
```

# PrintWriter 클래스

- PrintWriter 클래스는 다른 Stream과 다르게 Byte 출력 Stream과 문자 출력 Stream을 가지고 객체를 생성할 수 있는 클래스
- PrintWriter는 Reader가 없는 오직 출력을 위한 객체
- 자동 플러쉬(flush) 기능을 가지고 있음
- PrintWriter 클래스의 생성자에는 FileNotFoundException 예외를 발생하기 때문에 반드시 예외처리를 해야 함
- PrintWriter는 입력 Filter Stream이 존재하지 않는 대표적인 Stream 클래스
- **PrintStream과 PrintWriter는 유사함**
- PrintWriter는 PrintStream을 대신할 수 있도록 정의된 클래스이며 PrintWriter 클래스 활용을 권장
- printf(), println()등 문자열 단위의 출력이 필요하다면 반드시 PrintWriter를 사용

# PrintWriter 클래스

## ■ 생성자

생성자	설명
<code>PrintWriter(OutputStream out)</code>	자동 플러시 없이 <code>OutputStream</code> 객체로 <code>PrintWriter</code> 객체를 생성
<code>PrintWriter(OutputStream out, boolean autoFlush)</code>	자동 플러시를 할 수 있는 <code>PrintWriter</code> 객체를 생성
<code>PrintWriter(Writer out)</code>	자동 플러시 없이 <code>Writer</code> 객체로 <code>PrintWriter</code> 객체를 생성
<code>PrintWriter(Writer out, boolean autoFlush)</code>	행 플러시를 할 수 있는 <code>PrintWriter</code> 객체를 생성

# PrintWriter 클래스

## ■ 메소드

메소드	설명
append(char c)	c를 현재 스트림에 출력
<b>boolean</b> checkError()	Stream을 플래시 해, 그 에러의 상태를 체크
<b>void</b> close()	Stream을 닫음 출력되지 않는 데이터가 있으면 먼저 출력하고 스트림을 닫기
<b>void</b> flush()	Stream을 플래시 버퍼링되어 아직 기록되지 않은 데이터를 출력 스트림에 모두 출력
<b>void</b> print( <b>boolean</b> b)	<b>boolean</b> 값을 출력
<b>void</b> print(char c)	문자를 출력
<b>void</b> print(char[] s)	문자의 배열을 출력
<b>void</b> print( <b>double</b> d)	배정밀도 부동 소수점수(실수)를 출력
<b>void</b> print( <b>float</b> f)	부동 소수점수(실수)를 출력
<b>void</b> print( <b>int</b> i)	정수를 출력



# PrintWriter 클래스

## ■ 메소드

메소드	설명
<code>void print(long l)</code>	long 정수를 출력
<code>void print(Object obj)</code>	오브젝트를 출력
<code>void print(String s)</code>	스트링을 출력
<code>void println()</code>	행 단락 스트링을 기입하는 것에 따라, 현재의 행을 종료
<code>void println(boolean x)</code>	boolean 값을 출력 하고, 행을 종료
<code>void println(char x)</code>	문자를 출력 하고, 행을 종료
<code>void println(char[] x)</code>	문자의 배열을 출력 하고, 행을 종료
<code>void println(double x)</code>	배정밀도 부동 소수점수(실수)를 출력 하고, 행을 종료
<code>void println(float x)</code>	부동 소수점수(실수)를 출력 하고, 행을 종료
<code>void println(int x)</code>	정수값을 출력 하고, 행을 종료

# PrintWriter 클래스

## ■ 메소드

메소드	설명
<code>void println(long x)</code>	long 정수를 출력 하고, 행을 종료
<code>void println(Object x)</code>	Object를 출력 하고, 행을 종료
<code>void println(String x)</code>	String을 출력 하고, 행을 종료
<code>protected void setError()</code>	에러가 발생한 것을 나타냄
<code>void write(char[] buf)</code>	문자의 배열을 기입
<code>void write(char[] buf, int off, int len)</code>	문자의 배열의 일부를 기입
<code>void write(int c)</code>	단일의 문자를 기입
<code>void write(String s)</code>	스트링을 기입
<code>void write(String s, int off, int len)</code>	스트링의 일부를 기입
<code>format(String format, Object... args), printf(String format, Object... args)</code>	지정된 format으로 매개변수 args를 변환한 문자열을 출력

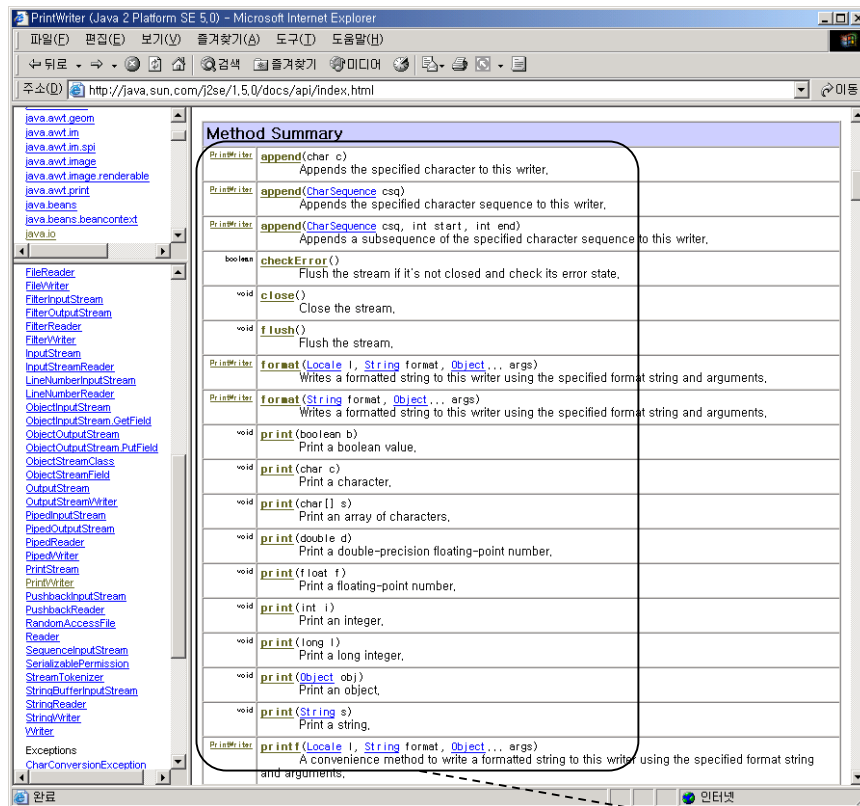
# PrintWriter 클래스

- print(), println() 메소드를 사용하여 출력을 편리하게 사용할 수 있도록 만든 클래스
- 콘솔창과 연결할 수 없기 때문에 File로 연결함

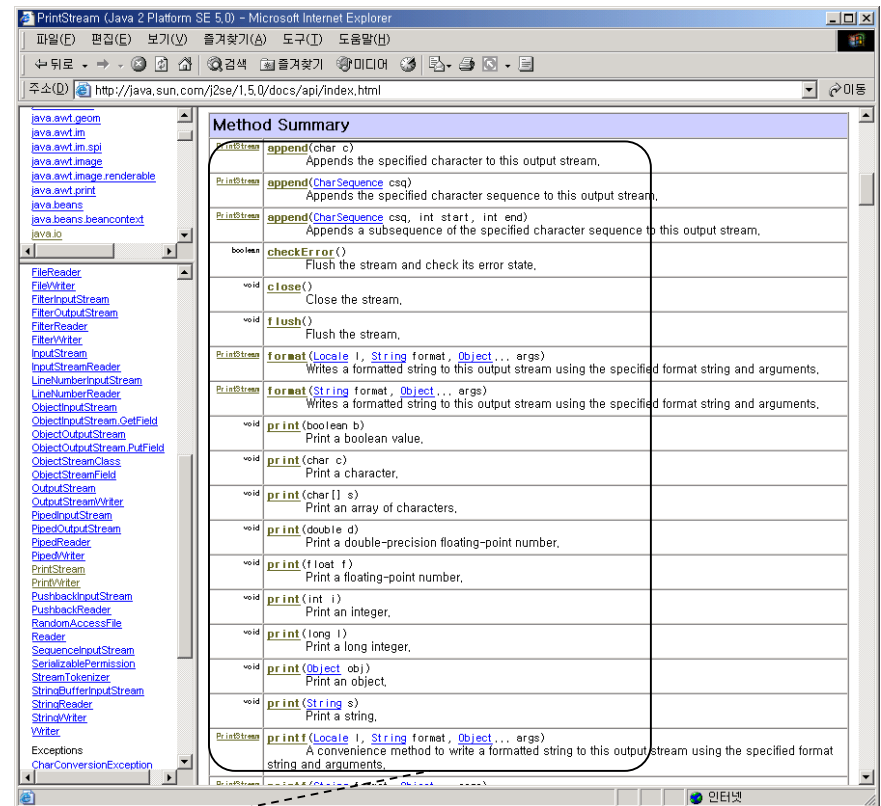
# PrintWriter 클래스

- Data를 Format해서 출력하는 클래스들
- PrintWriter 클래스와 PrintStream 클래스

[PrintWriter 클래스의 API 규격서]



[PrintStream 클래스의 API 규격서]



메소드의 기능과 사용 방법이 거의 동일합니다.

# PrintWriter 클래스

- 데이터를 포맷해서 파일로 출력하는 클래스
- 사용 방법
  - 다음과 같은 방법으로 PrintWriter 객체를 생성해서 파일을 연다

```
PrintWriter writer = new PrintWriter("output.txt");
```

↑  
생성자 안에서 이 파일을 엽니다.

# PrintWriter 클래스

## ■ 사용 방법

### ■ print, println, printf 메소드를 호출한다

```
writer.print(12);           // "12"라는 문자열을 출력  
writer.print(10000L);       // "10000"이라는 문자열을 출력
```

```
writer.print("12.5");       // "12.5"라는 문자열과 줄바꿈 문자를 출력  
writer.print("12.5\n");     // "12.5"라는 문자열과 줄바꿈 문자를 출력
```

```
writer.printf("%d년 %d월 %d일", 2006, 4, 19);  
// "2006년 4월 19일"이라는 문자열을 출력  
writer.printf("파이=%4.2f", Math.PI);  
// "파이=3.14"라는 문자열을 출력
```

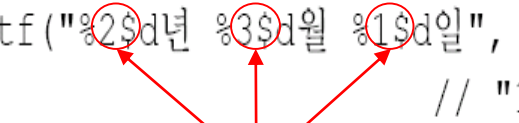
포맷 명세자(format specifier)

포맷 문자열(format string)

# PrintWriter 클래스

## ■ 아규먼트 인덱스(argument index)

```
writer.printf("%2$d년 %3$d월 %1$d일", 1, 1919, 3);  
// "1919년 3월 1일"이라는 문자열을 출력
```



아규먼트 인덱스(argument index)

```
writer.printf("%1$d는 16진수로 %1$x.", 100);  
// "100은 16진수로 64."라는 문자열을 출력
```

```
writer.printf("%1$tY년 %1$tm월 %1$td일", new GregorianCalendar());  
// 날짜를 "YYYY년 MM월 DD일" 포맷으로 출력
```

# PrintWriter 예제 1

```
public static void main(String[] args) {  
    String path = ".\\data\\sample.txt";  
  
    try{  
        PrintWriter writer = new PrintWriter(path);  
        writer.print("JAVA 공부가 재미 있다.");  
        writer.printf("%n 너무 너무 재미 있네%n");  
        writer.println("Hello World!");  
        writer.close( );  
        System.out.println("파일이 잘 만들어 졌습니다");  
    } catch(IOException e) {  
        System.err.println(e.getMessage());  
    }  
}
```



# PrintWriter 예제 2

```
public static void main(String[] args) {  
    PrintWriter output = new PrintWriter(System.out);  
  
    output.print("알기쉽게 해설한 자바! ");  
    output.printf("\n%s", "알기쉽게 해설한 자바!\n");  
    output.println('A');  
    output.println(500 + 500);  
    output.println(40000L);  
    output.println(12.34f);  
    output.println(12.34);  
    output.close();  
}
```

# PrintWriter 예제 3

데이터를 Format으로 출력하는  
PrintWriter 클래스

```
public static void main(String[] args) {  
    String path = ".\\data\\report.txt";
```

```
    try {  
        PrintWriter writer = new PrintWriter(path);  
        writer.println("    *** 성적표 ***    ");  
        writer.printf("작성일자 : %1$tY년 %1$tm월 %1$td일\n",  
                       new GregorianCalendar());  
        writer.println("*****");  
        writer.println("이름   국어   영어   수학   평균");  
        writer.println("*****");  
        writer.printf("%3s %5d %5d %5d %6.1f %n", "김지영", 92, 87, 95, 91.3f);  
        writer.printf("%3s %5d %5d %5d %6.1f %n", "박현식", 100, 90, 88, 92.7f);  
        writer.printf("%3s %5d %5d %5d %6.1f %n", "최민재", 75, 88, 85, 82.7f);  
        writer.println("*****");  
        System.out.println("파일로 출력이 완료되었습니다.");  
        writer.close();  
    } catch (IOException e){  
        System.err.println(e.getMessage());  
    }  
}
```

# PrintWriter 예제 4

```
public static void main(String[] args) {  
    String path = ".\\data\\test.txt";  
    Scanner keyboard = new Scanner(System.in);  
    try {  
        FileWriter outputStream = new FileWriter(path);  
        PrintWriter writer = new PrintWriter(outputStream);  
        while (true) {  
            System.out.print("성명 입력 : ");  
            String name = keyboard.next();  
            System.out.print("나이 입력 : ");  
            int age = keyboard.nextInt();  
            writer.printf("%s %d\\n", name, age);  
            System.out.print("계속 하시겠습니까 ? (Y/N) ");  
            char enter = keyboard.next().charAt(0);  
            if (enter == 'N' || enter == 'n')  
                break;  
        }  
    }  
}
```



Futuristic Innovator

京福大學校  
KYUNGBOK UNIVERSITY

# PrintWriter 예제 4

```
        writer.close();
    } catch (IOException e) {
        System.err.println(e.getMessage());
    }
}
```