Java ?? ?? ?? (????? ???, printf, format)

1. ????? ?????

> **이스케이프 시퀀스 (Escape Sequence)**는 문자열 내에서 특수한 동작 (줄바꿈 , 탭 , 따옴표 등)을 표현하기 위한 문자 조합이다 .
> Java에서는 \ 백슬래시로 시작한다 .

1.1 ?? ????? ???

이스케이프	의미	예시 출력
`\n`	줄바꿈	줄을 바꿈
`\t`	탭	띄어쓰기 (보통 4~8칸)
`\"`	큰따옴표 출력	`"그는 말했다: \"안녕\""`
`\'`	작은따옴표 출력	`'I\'m fine'`
`\\`	백슬래시 출력	`"C:\\Users\\Dain"`
`\r`	캐리지 리턴	커서를 맨 앞으로 이동 (거의 안 씀)
`\b`	백스페이스	글자 하나 삭제 (거의 안 씀)
`\f`	폼 피드	페이지 넘김 (거의 안 씀)

1.2 ?? ??

```
System.out.println("이름\t나이");
System.out.println("영희\t25");
System.out.println("그는 말했다: \"안녕하세요\"");
System.out.println("C:\\Program Files\\Java");
```

출력 결과 :

```
이름 나이
영희 25
그는 말했다: "안녕하세요"
C:\Program Files\Java
```

---

2. ??? ?? ?? ??

Java에서 문자열을 정해진 형식으로 출력하고 싶을 때 `System.out.printf()` 또는 `String.format()`을 사용한다.

2.1 ?? ??

```
System.out.printf("형식", 값1, 값2...);
String result = String.format("형식", 값1, 값2...);
```

2.2 ?? ?? ??

서식 문자	설명	예시 출력
`%s`	문자열	`"이름: %s"` → 이름 출력
`%d`	정수 (10진수)	`"나이: %d"`
`%f`	실수 (소수점 가능)	`"점수: %.2f"`
`%c`	문자	`"성별: %c"`
`%n`	줄바꿈 (OS별 호환)	`"다음 줄%n"`
`%%`	퍼센트 (%) 출력	`"100%% 완료"`

2.3 ???/?? ??

형식	설명
`%5d`	전체 5자리, 오른쪽 정렬
`%-5d`	전체 5자리, 왼쪽 정렬
`%05d`	앞자리를 0으로 채움 (총 5자리)

`%.2f`	소수점 둘째자리까지 출력
`%6.2f`	전체 6칸 중 소수점 2자리 포함

2.4 ?? ??

```
String name = "영희";
int age = 25;
double score = 93.756;

System.out.printf("이름: %s, 나이: %d%n", name, age);
System.out.printf("점수: %.2f%n", score);
System.out.printf("진행률: %d%%%n", 100);
```

출력 결과 :

```
이름: 영희, 나이: 25
점수: 93.76
진행률: 100%
```

3. ?? ???

항목	예시	설명
이스케이프 문자	`\n`, `\t`, `\\`	문자열 내부 제어
서식 문자 (printf)	`%d`, `%.2f`, `%s`	형식을 지정하여 출력

4. ?? ?? ?? (ANSI ??)

Java 자체는 콘솔 색상 기능을 제공하지 않지만 , 터미널에서 **ANSI** 코드 를 사용하면 출력 색을 변경할 수 있다 .

```
public class ColorExample {
    public static final String RED = "\u001B[31m";
    public static final String RESET = "\u001B[0m";
```

```
    public static void main(String[] args) {
        System.out.println(RED + "이 문장은 빨간색입니다." + RESET);
    }
}
```

색상	**ANSI** 코드
빨강	`\u001B[31m`
초록	`\u001B[32m`
노랑	`\u001B[33m`
파랑	`\u001B[34m`
리셋	`\u001B[0m`

---

5. ?? ? ?? ?? ??

Java에서는 `java.time.LocalDateTime`과 `DateTimeFormatter`를 사용해 날짜/시간을 원하는 형식으로 출력할 수 있다.

5.1 DateTimeFormatter ?? ??

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

LocalDateTime now = LocalDateTime.now();
DateTimeFormatter fmt = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
System.out.println("현재 시간: " + now.format(fmt));
```

5.2 ?? ?? ??

패턴	의미	예시
`yyyy`	연도	`2025`
`MM`	월	`05`
`dd`	일	`22`
`HH`	시간 (24시간제)	`14`

`hh`	시간 (12시간제)	`02`
`mm`	분	`07`
`ss`	초	`59`
`a`	오전/오후	`AM` / `PM`
`E`	요일	`Wed`

5.3 ??? ?? ?? ??

```
DateTimeFormatter customFmt = DateTimeFormatter.ofPattern("yyyy년 MM월 dd일 (E) a hh:mm");
String formatted = now.format(customFmt);
System.out.println("형식화된 시간: " + formatted);
```

6. ??? ?? ??

Java에는 기본 로그 시스템인 `java.util.logging.Logger`가 있다.

6.1 Logger ?? ??

```
import java.util.logging.*;

Logger logger = Logger.getLogger("MyLogger");

logger.info("정보 메시지");
logger.warning("경고 메시지");
logger.severe("에러 메시지");
```

6.2 ?? ???? ?? ??

```
String name = "다인";
int age = 25;
logger.info(String.format("사용자 이름: %s, 나이: %d", name, age));
```

6.3 ?? ?? ??

레벨	설명
`SEVERE`	심각한 오류
`WARNING`	경고
`INFO`	일반 정보
`CONFIG`	설정 관련 정보
`FINE`	디버깅 수준의 정보

6.4 ?? ?? ??????

- 로그 출력 위치 (콘솔 /파일)
- 출력 메시지 포맷 (시간 , 로그 레벨 , 메시지 내용 등)
- `logging.properties` 파일로 설정 변경 가능

Revision #2
Created 22 May 2025 05:10:47 by Dain
Updated 23 May 2025 01:06:36 by Dain