

2023.4.27

day16 파일

java_git 파일

I . II . III . IV . V . VI . VII .

<<요약>>

I . set_get class 분리

II . 시간 참조

III . sleep

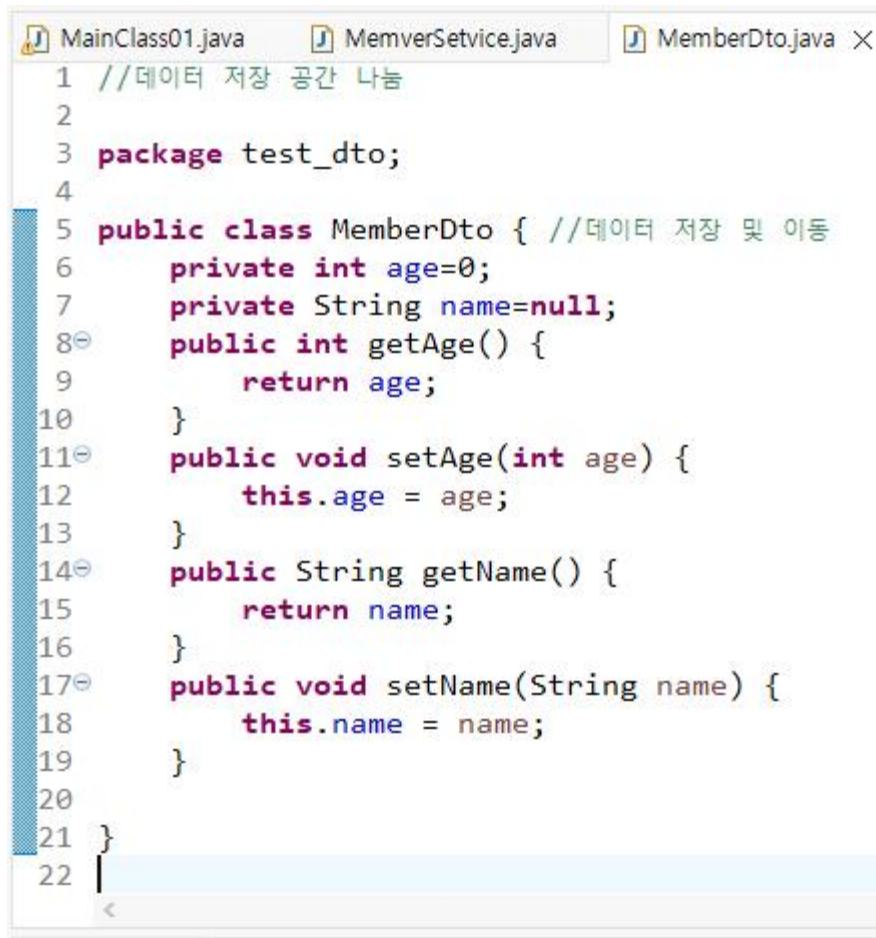
IV . ProcessBuilder

=====

I . set_get class 분리

*main Class - method Class - Data Class

(실행 공간 - 기능 공간 - 데이터 저장 공간)



```
1 //데이터 저장 공간 나눔
2
3 package test_dto;
4
5 public class MemberDto { //데이터 저장 및 이동
6     private int age=0;
7     private String name=null;
8     public int getAge() {
9         return age;
10    }
11    public void setAge(int age) {
12        this.age = age;
13    }
14    public String getName() {
15        return name;
16    }
17    public void setName(String name) {
18        this.name = name;
19    }
20
21 }
22
```

```
MainClass01.java  MemverSetvice.java X  MemberDto.java  MainClass01.java
1 //메소드 - 기능 나눔
2 package test_dto;
3
4 import java.util.Scanner;
5
6 public class MemverSetvice {
7
8     Scanner input = new Scanner(System.in);
9     MemberDto dto = new MemberDto();
10
11     public void inputName(){
12         System.out.println("이름 입력");
13         String name = input.next();//지역변수
14         dto.setName(name);//dto name에 입력 받은 name에 추가
15     }
16
17     public void inputAge() {
18         System.out.println("나이 입력");
19         //dto.setAge(age) = input.nextInt();아래와 같은 내용
20         dto.setAge(input.nextInt());
21     }
22
23     public void print() {
24         System.out.println("이름 : "+dto.getName()); // 초기화 필요
25         int age = dto.getAge();//위아래 둘다 가능
26         System.out.println("만 나이 : "+age);
27     }
28 }
```

Ⅱ. 시간 참조

```
1. currentTimeMillis();
```

```
* long t = System.currentTimeMillis();
```

- long type

- 형변환 가능, 값이 다를 수 있음 `int t = (int)System.currentTimeMillis();`

2. Date

```
Date d=new Date();
```

:요일 - 일 - 시 - 분 - 초

3. SimpleDateFormat

- 포맷 변경

```
* SimpleDateFormat s =
```

```
new SimpleDateFormat("yyyy-MM-dd일 aaa hh시 mm분 ss초");
```

- 형식 표현 aaa(오전 오후)

```
* String s_t = s.format(d);
```

```
MainClass01.java  MemberService.java  MemberDto.java  MainClass01.java X
1 package time_ex;
2
3 import java.text.SimpleDateFormat;
4 import java.util.Date;
5
6 public class MainClass01 {
7
8     public static void main(String[] args) {
9         long t = System.currentTimeMillis(); //long type
10        //정변화 가능, 값이 다를 수 있음 int t = (int)System.currentTimeMillis();
11        System.out.println(t); //현재 시간 초단위로 가져옴
12
13
14        Date d=new Date();
15        System.out.println(d); // 요일 - 월 - 일 - 시 -분 -초
16
17        //포맷 변경
18        SimpleDateFormat s=
19            new SimpleDateFormat("yyyy-MM-dd일 aaa hh시 mm분 ss초");//형식 표현 aaa(오전 오후)
20        String s_t = s.format(d);
21        System.out.println(s_t);
22
23    }
24 }
```

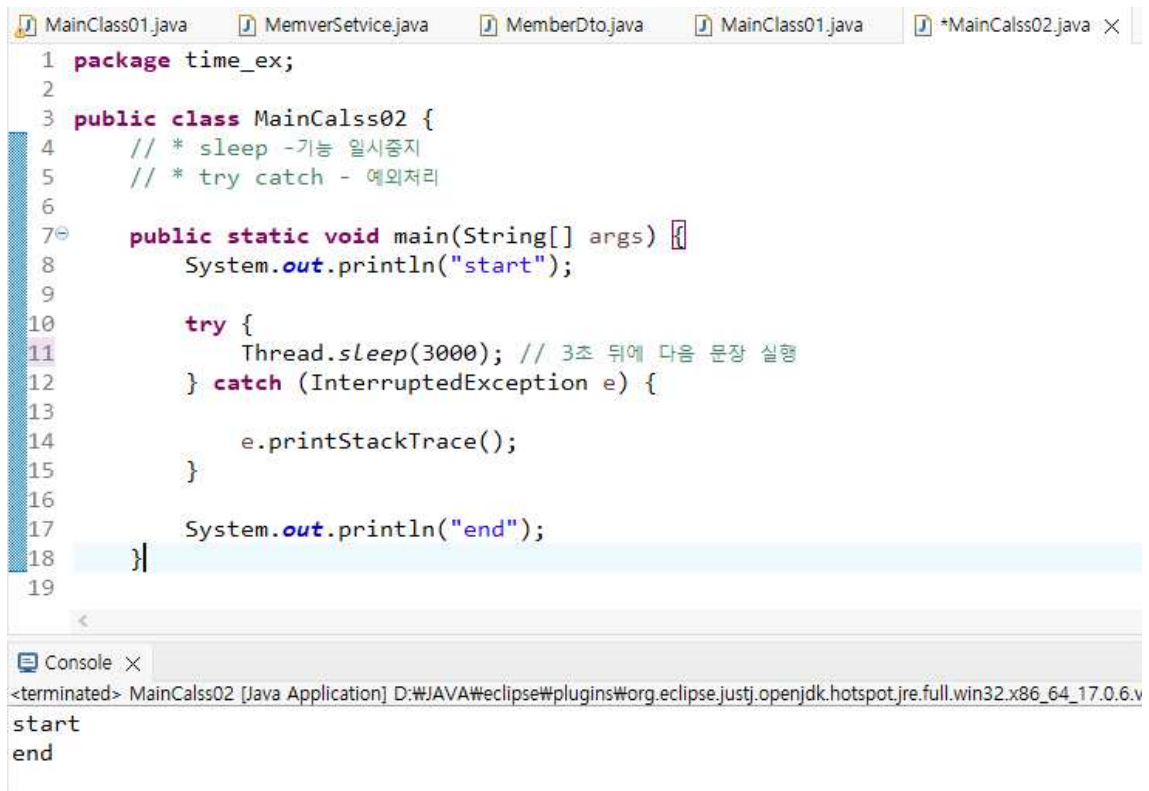
Console X

```
<terminated> MainClass01 (5) [Java Application] D:\JAVA\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\java.exe
1682556651730
Thu Apr 27 09:50:51 KST 2023
2023-04_27일 오전 09시 50분 51초
```

III. sleep

: 기능 일시 중지

*`long start = System.currentTimeMillis();`



```
1 package time_ex;
2
3 public class MainCalss02 {
4     // * sleep -기능 일시중지
5     // * try catch - 예외처리
6
7     public static void main(String[] args) {
8         System.out.println("start");
9
10        try {
11            Thread.sleep(3000); // 3초 뒤에 다음 문장 실행
12        } catch (InterruptedException e) {
13
14            e.printStackTrace();
15        }
16
17        System.out.println("end");
18    }
19 }
```

Console X

<terminated> MainCalss02 [Java Application] D:\#JAVA#eclipse#plugins#org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v

start

end

IV. ProcessBuilder

:윈도우 명령어 실행

*`ProcessBuilder pro = new ProcessBuilder();`

String s = "calc";

pro.command(s);

V. 패키지

*default - 다른 패키지 참조 X(예 - public 없는 class)

*패키지 참조

```

MainClass01.java  MemverSetvice.java  MemberDto.java  MainClass01.java  *MainCalss02.java X
1 package time_ex;
2
3 public class MainCalss02 {
4     // * sleep -기능 일시중지
5     // * try catch - 예외처리
6
7     public static void main(String[] args) {
8         System.out.println("start");
9
10        try {
11            Thread.sleep(3000); // 3초 뒤에 다음 문장 실행
12        } catch (InterruptedException e) {
13
14            e.printStackTrace();
15        }
16
17        System.out.println("end");
18    }
19

```

```

Console X
<terminated> MainCalss02 [Java Application] D:\JAVA#eclipse#plugins#org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v
start
end

```

```

MainClass01.java  MemverSetvice.java  MemberDto.java  MainClass01.java  MainCalss02.java
1 package time_ex;
2
3 import java.io.IOException;
4
5 public class MainClass03 {
6     public static void main(String[] args) {
7         ProcessBuilder pro = new ProcessBuilder();
8         String s = "calc";
9         pro.command(s); //calc : 계산기 등 윈도우 명령어 실행
10
11        try {
12            pro.start();
13        } catch (IOException e) {
14            e.printStackTrace();
15        } //실행
16    }
17 }
18

```

VI. constructor 생성자 & 메소드 오버로딩

```
MainClass.java ×
3 // *
4 constructor(생성자)
5 - 초기화 목적
6 - 객체를 생성할 때 자동으로 호출된다
7 - 클래스 이름과 동일한 메소드 이름을 가지며 반환타입, 값은 없다.
8 */
9
10 class TestClass01{
11     public TestClass01(int age) {
12         System.out.println("생성자 호출");
13     }
14
15     public void test() {
16         System.out.println("test");
17     }
18 }
19
20 public class MainClass {
21     public static void main(String[] args) {
22         TestClass01 t = new TestClass01(100);
23         t.test();
24     }
25 }
26 }
27
Console ×
```

* constructor(생성자)

- 초기화 목적
- 객체를 생성할 때 자동으로 호출된다
- 클래스 이름과 동일한 메소드 이름을 가지며 반환타입, 값은 없다.

//메소드 오버로딩

```
class TestClass01{
    int age;
    public TestClass01(int age) {

        System.out.println(age + "생성자 호출");
        this.age = age;
    }
    public TestClass01(){ // 에러 줄이는 법
        System.out.println("기본 생성자");
    } //매개변수 전달 오류 줄이기
    public void test() {
        System.out.println("test");
    }
    public TestClass01(int a, int b){ // 에러 줄이는 법
        System.out.println("2개 받아옴");
    }
}
```



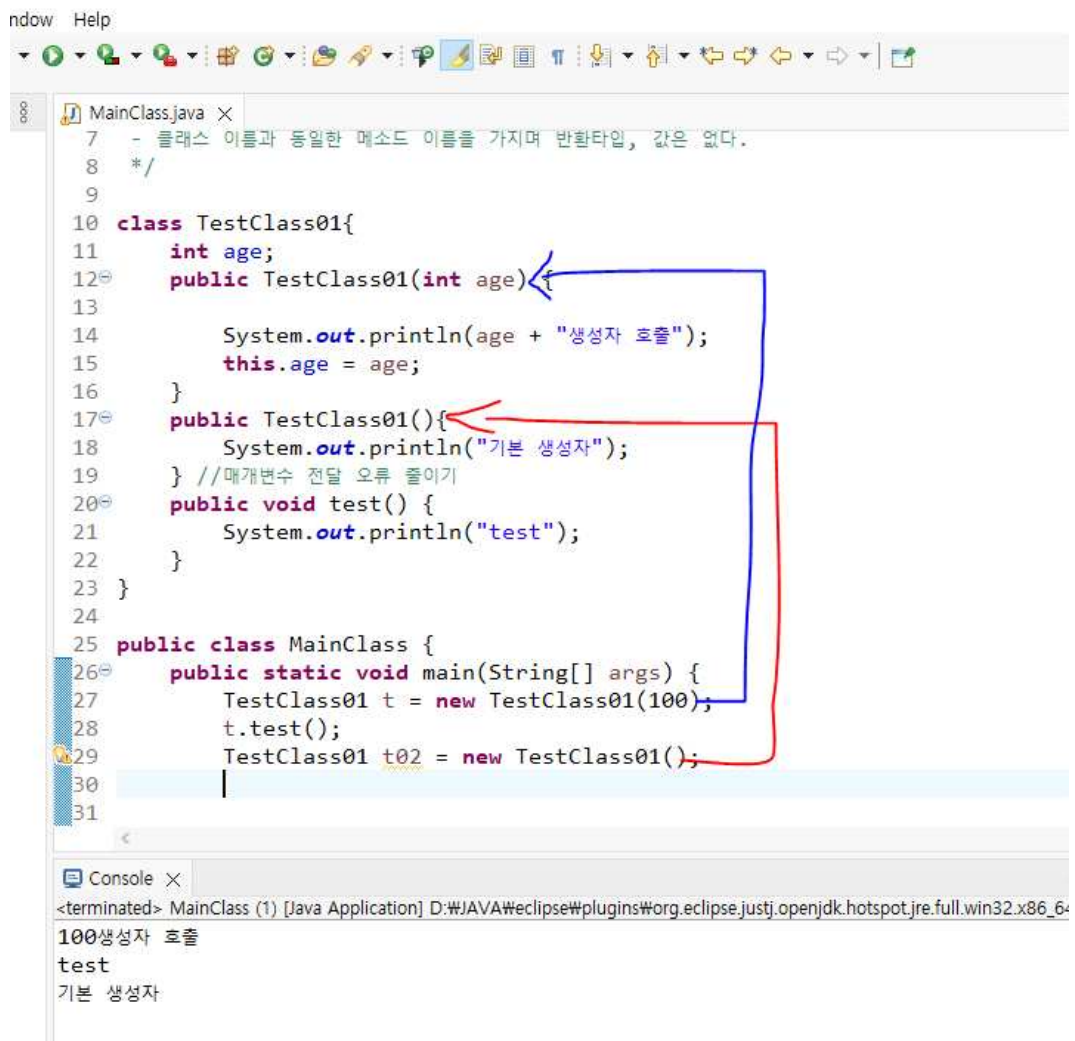
```

    }
}

public class MainClass {
    public static void main(String[] args) {
        TestClass01 t = new TestClass01(100);
        t.test();
        TestClass01 t02 = new TestClass01();
        String s = new String();
        String s1 = new String("aaa");
        TestClass01 t03 = new TestClass01(10,20);

    }
}

```



The screenshot shows the Eclipse IDE with a Java project. The editor displays the following code:

```

7  - 클래스 이름과 동일한 메소드 이름을 가지며 반환타입, 값은 없다.
8  */
9
10 class TestClass01{
11     int age;
12     public TestClass01(int age){
13
14         System.out.println(age + "생성자 호출");
15         this.age = age;
16     }
17     public TestClass01(){
18         System.out.println("기본 생성자");
19     } //매개변수 전달 오류 줄이기
20     public void test() {
21         System.out.println("test");
22     }
23 }
24
25 public class MainClass {
26     public static void main(String[] args) {
27         TestClass01 t = new TestClass01(100);
28         t.test();
29         TestClass01 t02 = new TestClass01();
30
31

```

Red and blue arrows highlight the constructor calls in the main method. The console output shows the results of the program execution:

```

<terminated> MainClass (1) [Java Application] D:\JAVA\ eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
100생성자 호출
test
기본 생성자

```