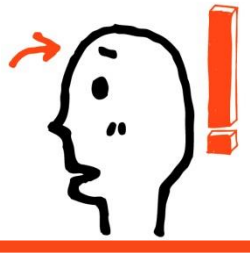
Android Java

5장 메소드와 변수의 스코프

메소드와 변수의 스코프

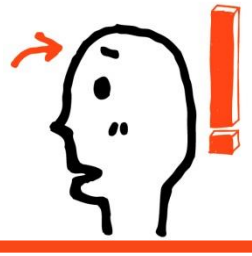


01 메소드에 대한 이해와 메소드의 정의

02 변수의 스코프

03 메소드의 재귀호출

main 메소드, 아는 것과 모르는 것

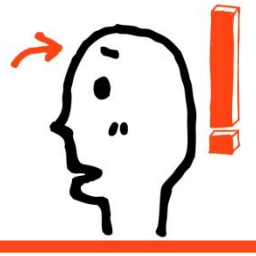


- 아는 것
  - 메소드 이름은 main
  - 메소드 중괄호 내에 존재하는 문장들이 위에서 아래로 순차적 실행
- 모르는 것
  - public, static, void
  - 왜? 항상 main인가?
  - String[] args

```
public static void main(String[] args)
{
    int num1=5, num2=7;
    System.out.println("5+7="+(num1+num2));
}
```

자바 프로그램의 시작은 main이라는 이름의 메소드를 실행하는 데서부터 시작한다!

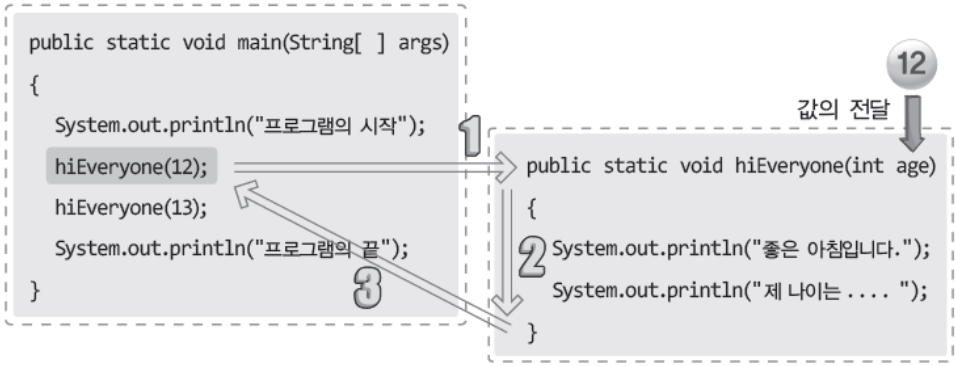
다른 이름의 메소드 만들기



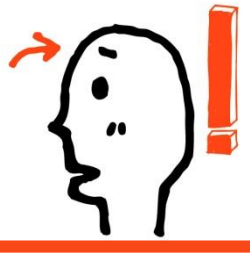
```
class MethodDefAdd
{
    public static void main(String[] args)
    {
        System.out.println("프로그램의 시작");
        hiEveryone(12);
        hiEveryone(13);
        System.out.println("프로그램의 끝");
    }
    public static void hiEveryone(int age)
    {
        System.out.println("좋은 아침입니다.");
        System.out.println("제 나이는 "+ age+"세입니다.");
    }
}
```

실행 결과

```
프로그램의 시작
좋은 아침입니다.
제 나이는 12세입니다.
좋은 아침입니다.
제 나이는 13세입니다.
프로그램의 끝
```



매개변수가 두 개인 형태의 메소드



```
class Method2Param
{
    public static void main(String[] args)
    {
        double myHeight=175.9;
        hiEveryone(12, 12.5);
        hiEveryone(13, myHeight);
        byEveryone();
    }
    public static void hiEveryone(int age, double height)
    {
        System.out.println("제 나이는 "+ age+"세 입니다.");
        System.out.println("저의 키는 "+ height+"cm 입니다.");
    }
    public static void byEveryone()
    {
        System.out.println("다음에 뵙겠습니다.");
    }
}
```

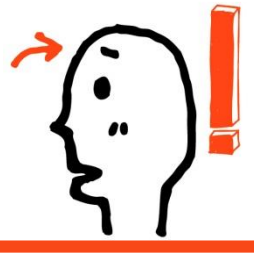
전달 순서대로 저장

전달되는 것 없음

실행 결과

```
제 나이는 12세 입니다.
저의 키는 12.5cm 입니다.
제 나이는 13세 입니다.
저의 키는 175.9cm 입니다.
다음에 뵙겠습니다.
```

값을 반환하는 메소드



```
class MethodReturns
{
    public static void main(String[] args)
    {
        int result=adder(4, 5);
        System.out.println("4와 5의 합 : " + result);
        System.out.println("3.5의 제곱 : " + square(3.5));
    }
    public static int adder(int num1, int num2)
    {
        int addResult=num1+num2;
        return addResult;
    }
    public static double square(double num)
    {
        return num*num;
    }
}
```

값을 반환하지 않겠다.

int형 데이터를 반환하겠다.

double형 데이터를 반환하겠다.

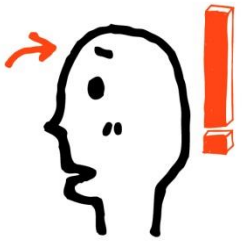
```
int result = adder(4, 5) ;
        ↓
int result = 9 ;
```

값의 반환이 의미하는 바

실행결과

```
4와 5의 합 : 9
3.5의 제곱 : 12.25
```

키워드 return이 지니는 두 가지 의미



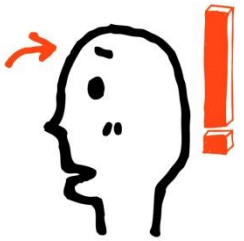
```
class OnlyExitReturn
{
    public static void main(String[] args)
    {
        divide(4, 2);
        divide(6, 2);
        divide(9, 0);
    }
    public static void divide(int num1, int num2)
    {
        if(num2==0)
        {
            System.out.println("0으로는 값을 나눌 수 없습니다.");
            return;
        }
        System.out.println("나눗셈 결과 : " + (num1/num2));
    }
}
```

실행결과

```
나눗셈 결과 : 2
나눗셈 결과 : 3
0으로는 값을 나눌 수 없습니다.
```

값의 반환, 메소드의 종료, 이렇게 두 가지의 의미를 지님

가시성 : 여기서는 저 변수가 보여요.



```
class LocalVariable
{
    public static void main(String[] args)
    {
        boolean scope=true;
        if(scope)
        {
            int num=1;
            num++;
            System.out.println(num);
        }
        else
        {
            int num=2;
            System.out.println(num);
        }
        simple();
    }
    public static void simple()
    {
        int num=3;
        System.out.println(num);
    }
}
```

num=1의 유효범위

num=2의 유효범위

변수 scope의 유효범위

num=3의 유효범위

선언된 지역을 벗어나면 변수는 자동 소멸된다.

```
for( int num=0 ; num<5; num++)
{
   /*추가적인
     변수 num 선언 불가 지역 */
}
```

변수 num의 접근 가능지역

```
public static void myFunc( int num )
{
   /*추가적인
     변수 num 선언 불가 지역 */
}
```

변수 num의 접근 가능지역