



메소드로 코드 간추리기

이것이 C#이다



Contents

- ❖ 메소드란?
- ❖ Return에 대하여
- ❖ 매개 변수에 대하여
- ❖ 참조에 의한 매개 변수 전달
- ❖ 메소드의 결과를 참조로 반환하기
- ❖ 출력 전용 매개 변수
- ❖ 메소드 오버로딩
- ❖ 가변길이 매개 변수
- ❖ 명명된 매개 변수
- ❖ 선택적 매개 변수
- ❖ 로컬 함수



6.1 메소드란? (1)

❖ 일련의 코드를 하나의 이름 아래 묶은 것

- 메소드, 함수(Function), 프로시저(Procedure), 서브루틴, 서브 프로그램

❖ 메소드 선언 형식

```
class Calculator
{
    public static int Plus( int a, int b )
    {
        Console.WriteLine("Input : {0}, {1}", a, b);

        int result = a + b;
        return result;
    }
}
```

```
}
```



6.1 메소드란? (2)

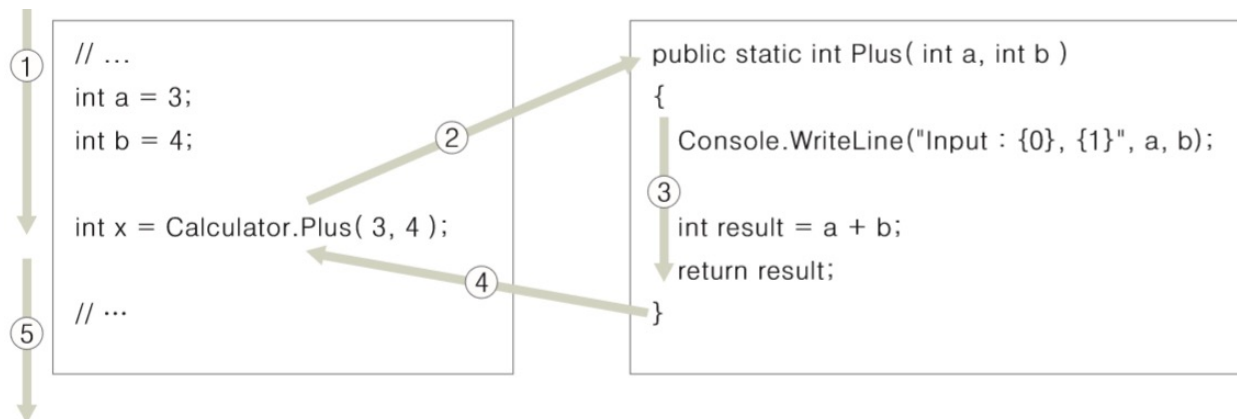
❖ 메서드 호출 – 메서드의 이름을 불러 주는 동작

int a = 3
매개 변수 목록

```
int x = Calculator.Plus( 3, 4 ); // x는 7  
int y = Calculator.Plus( 5, 6 ); // y는 11  
int z = Calculator.Plus( 7, 8 ); // y는 15
```

7
메소드의 결과

❖ 메서드 호출 시 일어나는 프로그램 흐름의 변화



❖ 데모 예제 – Calculator



6.2 return에 대하여

- ❖ 점프문의 한 종류
- ❖ 프로그램의 흐름을 호출자에게로 돌려놓음.
- ❖ 메소드의 어느 위치에서나 호출 가능
- ❖ 사용 예

```
void PrintProfile(string name, string phone)
{
    if (name == "")
    {
        Console.WriteLine("이름을 입력해주세요.");
        return;
    }

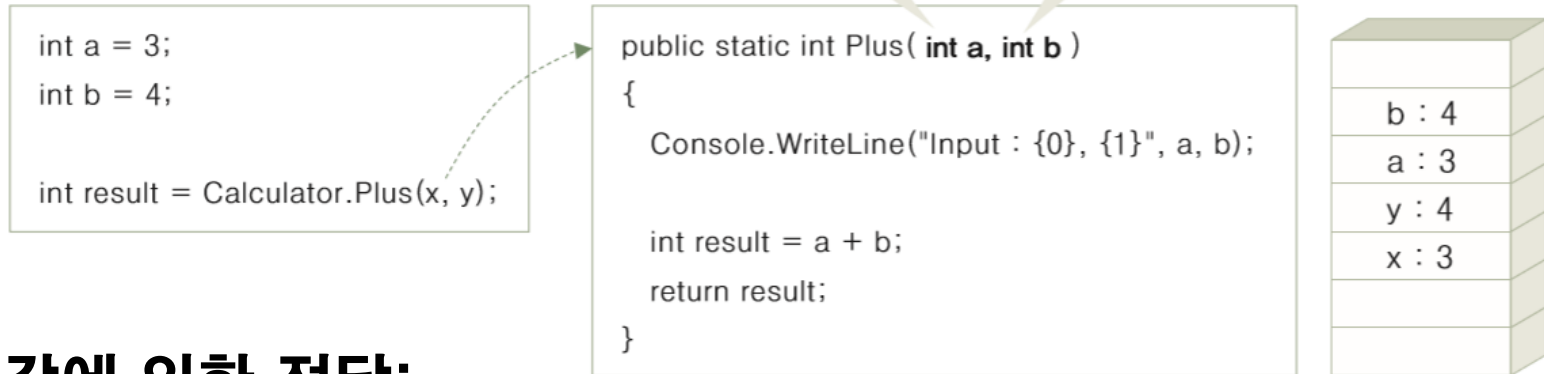
    Console.WriteLine( "Name:{0}, Phone:{1}", name, phone );
}
```

- ❖ 데모 예제 - Return



6.3 매개 변수에 대하여

❖ 매개변수가 전달되는 과정



❖ 값에 의한 전달:

■ 메소드 호출 시 데이터를 복사해 매개 변수에 전달

```
static void Main(string[] args)
{
    int x = 3;
    int y = 4;

    Swap(x, y);
}
```

❖ 데모 예제 – SwapByValue



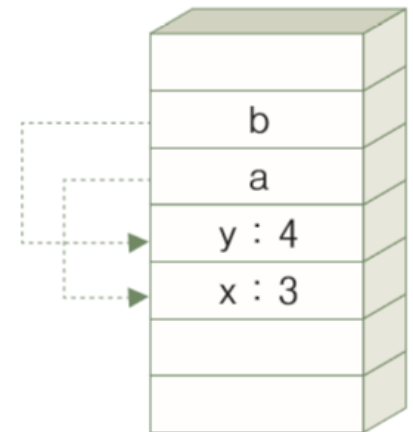
6.4 참조에 의한 매개 변수 전달

- ❖ 매개 변수로 직접 원본 변수의 값을 바꾸는 방법
- ❖ ref 키워드

```
int x = 3;  
int y = 4;  
  
Swap(ref x, ref y);
```

```
int a = 3;  
int b = 4;  
  
Swap( ref x, ref y );
```

```
static void Swap( ref int a, ref int b )  
{  
    int temp = b  
    b = a;  
    a = temp;  
}
```



- ❖ 데모 예제 – SwapByRef



6.5 메소드의 결과를 참조로 반환하기

❖ 메소드 호출자가 반환 결과를 참조로 다룰 수 있다.

❖ 선언 방법

- ref 한정자로 메소드 선언
- return문이 반환하는 변수에 ref 키워드 사용

```
SomeClass obj = new SomeClass();  
int result = obj.SomeMethod();
```

값으로 반환받고자 할 때는 어느 때와
다름없이 메소드를 호출하면 됩니다.

```
SomeClass obj = new SomeClass();  
ref int result = ref obj.SomeMethod(); // result는 참조 지역 변수입니다.
```

❖ 데모 예제 - RefReturn



6.6 출력 전용 매개 변수

❖ 두 가지 이상의 결과가 필요한 메소드

❖ ref 키워드를 이용한 방법

```
void Divide( int a, int b, ref int quotient, ref int remainder )  
{  
    quotient = a / b;  
    remainder = a % b;  
}
```

❖ out 키워드를 이용한 방법 ← 권장

■ 컴파일러를 통해 결과를 할당하지 않는 버그를 만들 가능성 제거

```
void Divide( int a, int b, out int quotient, out int remainder )  
{  
    quotient = a / b;  
    remainder = a % b;  
}
```

❖ 데모 예제 - UsingOut



6.7 메소드 오버로딩

- ❖ 하나의 메소드 이름에 여러 개의 구현을 올리는 것
- ❖ 매개 변수의 수와 형식을 분석해 호출할 메소드 결정

```
int Plus(int a, int b)
{
```

```
int    result1 = Plus( 1, 2 );
double result2 = Plus( 3.1, 2.4 );
```

```
{
    return a + b;
}
```

int Plus(int, int)를 호출합니다.

double Plus(double, double)를 호출합니다.

- ❖ 이름에 대한 고민을 덜어준다.
- ❖ 코드의 일관성 제공
- ❖ 데모 예제 - Overloading



6.8 가변길이 매개 변수

- ❖ 형식은 같으나 매개 변수의 개수만 유연하게 달라질 수 있는 경우에 적합.
- ❖ params 키워드와 배열 이용

```
int Sum( params int[] args )  
{  
    int sum = 0;  
  
    for(int i=0; i<args.Length; i++)  
    {  
        sum += args[i];  
    }  
    return sum;  
}
```

Sum() 메소드에 입력한 모든
메소드는 args 배열에 담깁니다.

- ❖ 데모 예제 – UsingParams



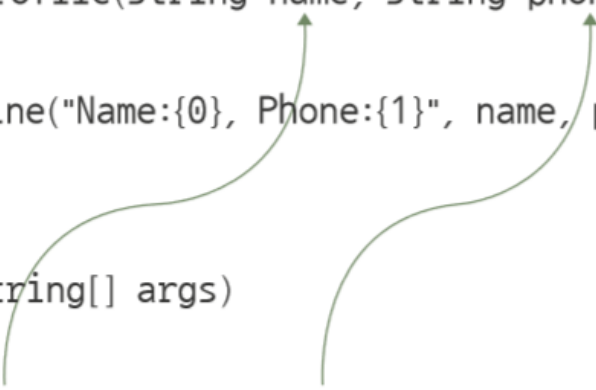
6.9 명명된 매개 변수

❖ 메소드를 호출할 때 매개 변수의 이름에 근거해서 데이터를 할당하는 기능

■ 매개 변수 이름: 값

```
static void PrintProfile(string name, string phone)
{
    Console.WriteLine("Name:{0}, Phone:{1}", name, phone);
}

static void Main(string[] args)
{
    PrintProfile(name : "박찬호", phone : "010-123-1234");
}
```



❖ 가독성에 도움

❖ 데모 예제 - NamedParameter



6.10 선택적 매개 변수

- ❖ 메소드의 매개 변수는 기본 값을 가질 수 있다.
 - 필요에 따라 데이터를 할당하거나 할당하지 않을 자유
 - 위치-필수 매개 변수(있다면) 다음.

❖ 사용 예

```
void MyMethod_0( int a = 0 )  
{  
    Console.WriteLine( "{0}", a );  
}  
  
void MyMethod_1( int a, int b = 0 )  
{  
    Console.WriteLine( "{0}, {1}", a, b );  
}  
  
void MyMethod_2( int a, int b, int c = 10, int d = 20 )  
{  
    Console.WriteLine( "{0}, {1}, {2}, {3}", a, b, c, d );  
}
```

- ❖ 모호함 발생 가능성 → 명명된 매개 변수 활용
- ❖ 데모 예제- OptionalParameter



6.11 로컬 함수

- ❖ 메소드 내에 선언하고, 그 안에서만 사용하는 특별한 함수
 - 클래스의 멤버가 아니 라서 함수라고 명명
 - 자신이 존재하는 지역에 선언된 변수 사용
 - 메소드 밖에서는 다시 쓸 일 없는 반복적인 작업을 하나의 이름 아래 묶어 놓는 데 제격

❖ 사용 예

```
public void SomeMethod()
{
    int count = 0;
    SomeLocalFunction(1, 2);
    SomeLocalFunction(3, 4);

    void SomeLocalFunction(int a, int b)
    {
        // Do Some Work
        Console.WriteLine($"count : {++count}");
    }
}
```

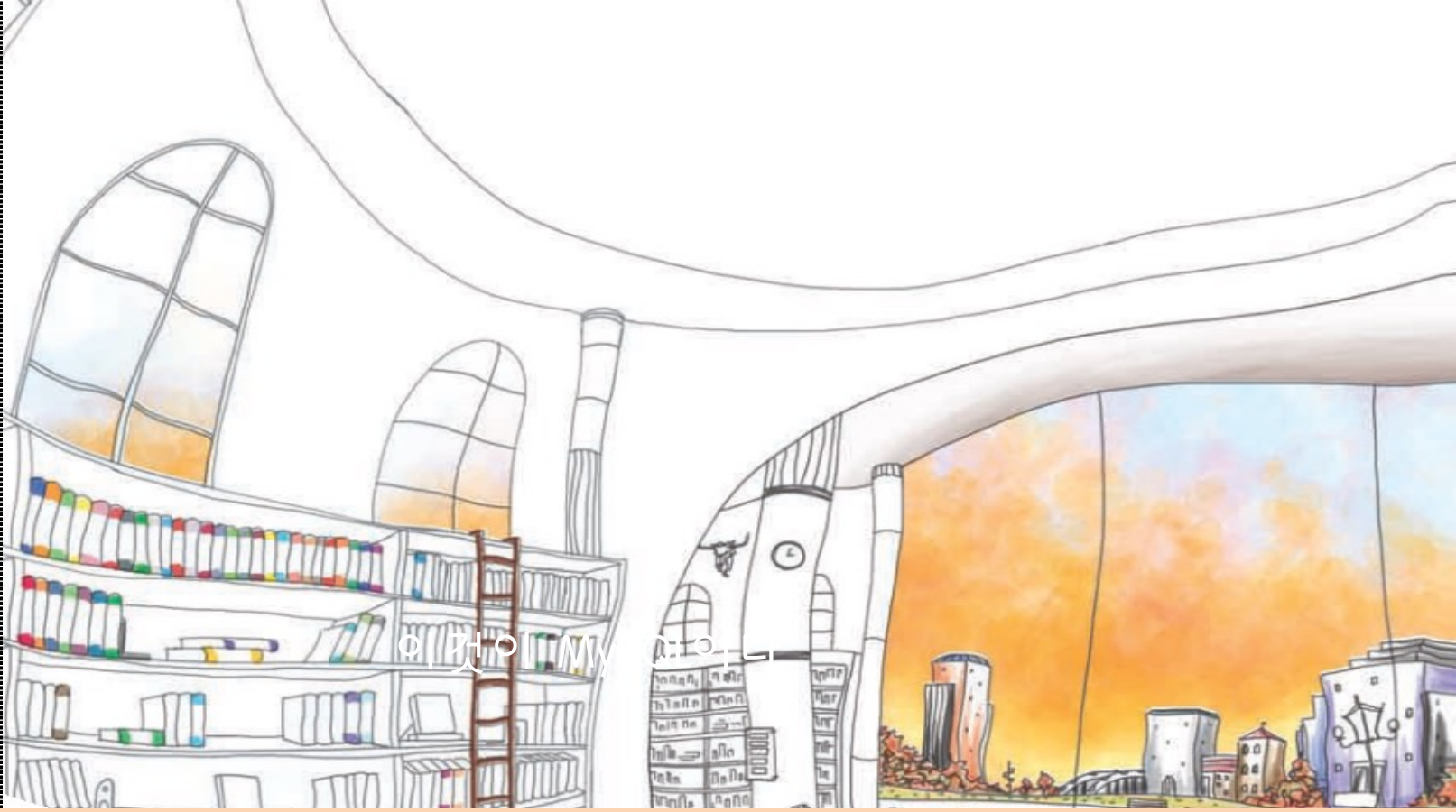
메소드 선언

로컬 함수 호출

로컬 함수 선언

로컬 함수는 자신이 소속한 메소드의 지역 변수를 사용할 수 있습니다.

❖ 데모 예제 - LocalFunction



Thank You !

이것이 C#이다

