Chapter04

JAVA 기초 문법

개요

1 학습목표

- ✓ **Java** 기본문법 구조에 대해서 알아본다.
- ✓ **Java**를 구성하는 변수와 데이터 타입, 연산자, 제어문, 클래스 및 객체에 대해서 알아본다.

2 학습내용

- ✓ 기본문법 구조
- ✓ 변수와 데이터 타입
- ✓ 연산자
- ✓ 제어문
- ✓ 클래스 및 객체
- ✓ **import** 및 주석

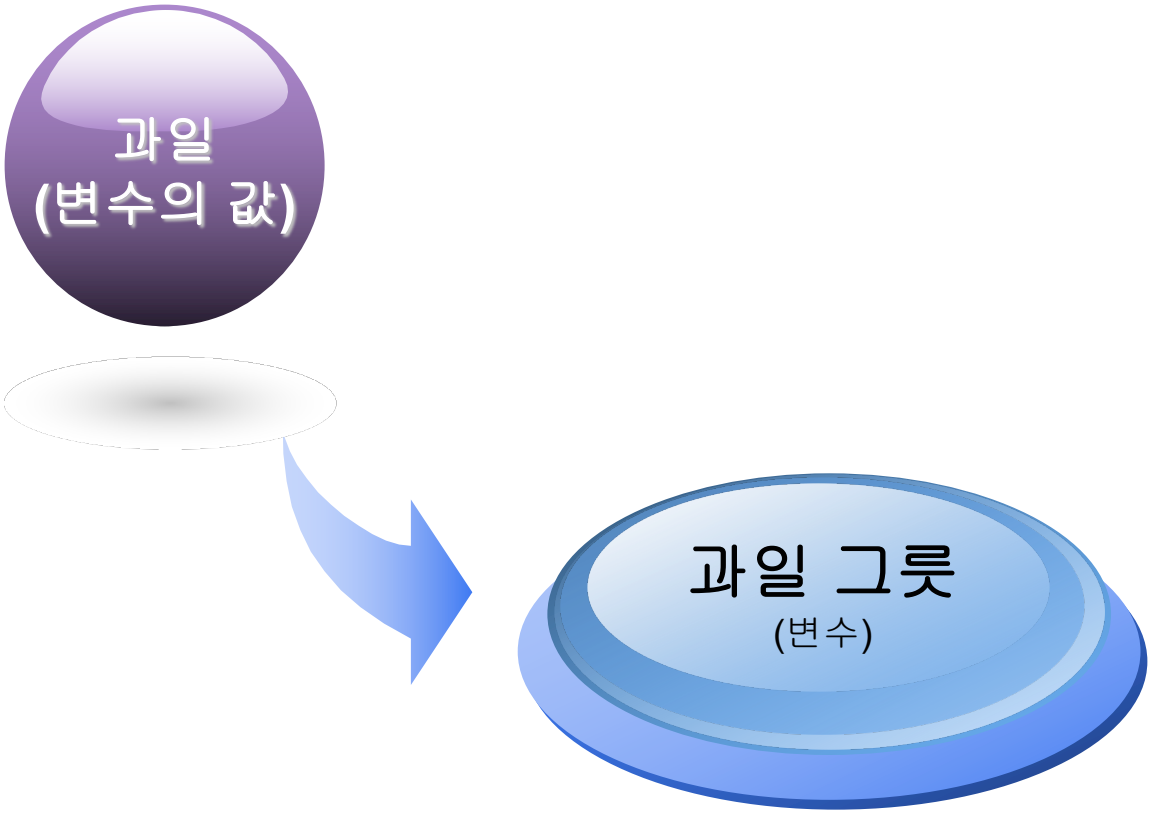
❖ Java class의 일반적인 형태

- source/ch04/Grammar.java

```
01 : package ch04;
02 :
03 : public class Grammar {
04 :
05 :  public String name = "JSPStudy";
06 :
07 :  public void jspStudy(int year) {
                int year는 매개변수입니다. 메소드가 결과값을 계산하기 위한 입력값을 외부로부터 전달받을 때 기술합니다.
08 :          System.out.println("변수 name은 " + name + "입니다.");
09 :          System.out.println("변수 year는 " + year + "입니다.");
10 :          System.out.println(name + year);
11 :  }
12 :
13 :  public static void main(String[] args) {
14 :          int year = 2013;
15 :          Grammar g = new Grammar();
16 :          g.jspStudy(year);
17 :  }
18 : }
```

변수와 데이터 타입

❖ 변수



변수와 데이터 타입

❖ 변수 선언

```
String name;
String jsp;
```

❖ 변수 할당

name = "JSPStudy";

❖ 변수 참조

```
System.out.print(name);
```

❖ 변수 선언, 할당, 참조 테스트를 위한 코드를 작성

- source/ch04/Variable.java

결과

변수 name은 JSPStudy입니다.
변수 year는 2013입니다.
JSPStudy2013

❖ 배열의 선언과 크기 지정

```
char a[], b = new char[5]; // 'a[]'는 배열이 되고 'b'는 char타입 일반 변수가 됩니다.
char[] a, b = new char[5]; // a, b 모두 배열로 선언됩니다.
```

❖ 선언과 동시에 배열에 값을 할당

```
char a[] = { '가', '나', '다', '라' };
```

❖ 배열을 활용한 코드를 작성

- source/ch04/Array.java

결과

가나다라

❖ 데이터 타입

- 정수형

[표 4-1] 정수형 데이터

형식	크기	범위
byte	1byte	-128 ~ 127
short	2byte	-32768 ~ 32767
int	4byte	-2147483648 ~ 2147483647
long	8byte	-9223372036854775808 ~ 9223372036854775807

❖ 데이터 타입

- 실수형

[표 4-2] 실수형 데이터

형식	크기	범위
float	4byte	-3.4E38 ~ 3.4E38
double	8byte	-1.7E308 ~ 1.7E308

변수와 데이터 타입

❖ 데이터 타입

- 논리형, 문자형

[표 4-3] 논리형, 문자형 데이터

형식	크기	범위
boolean	1byte	true 또는 false
char	2byte	유니코드(Unicode)

❖ 산술연산자와 증감연산자

[표 4-4] 산술연산자와 증감 연산자

	연산자	예시	의미
산술연산자	+	a+b	a와 b를 더합니다.
	−	a−b	a에서 b를 뺍니다.
	*	a*b	a와 b를 곱합니다.
	/	a/b	a를 b로 나눕니다.
	%	a%b	a를 b로 나눴을 때의 나머지를 구합니다.
증감연산자	++	++a a++	a를 1 증가시킨 후 참조합니다. a를 먼저 참조한 후 1 증가시킵니다.
	—	−−a a−−	a를 1 감소시킨 후 참조합니다. a를 먼저 참조한 후 1 감소시킵니다.

연산자

- 산술연산자와 증감연산자 실습을 위해 코드를 작성합니다.
  - source/ch04/Arithmetic.java

```
결과
------------------------
산술연산자
a + b는 8입니다.
a - b는 2입니다.
a * b는 15입니다.
a / b는 1입니다.
a / b의 나머지는 2입니다.
```

❖ 비교연산자와 대입연산자

[표 4-5] 비교연산자와 대입연산자

	연산자	예시	의미
비교연산자	〈	a 〈 b	a가 b보다 작을 경우 true를 반환합니다.
	〈=	a 〈= b	a가 b보다 작거나 같을 경우 true를 반환합니다.
	〉	a 〉 b	a가 b보다 클 경우 true를 반환합니다.
	〉=	a 〉= b	a가 b보다 크거나 같을 경우 true를 반환합니다.
	!=	a != b	a가 b가 같지 않을 때 true를 반환합니다.
	=	a == b	a가 b가 같을 때 true를 반환합니다.
대입연산자	==	a = b	b의 값을 a에 대입합니다.
	+=	a += b	b의 값을 a에 더하고 대입합니다.
	-=	a -= b	a에서 b의 값을 빼고 대입합니다.
	*=	a *= b	a에 b의 값을 곱하고 대입합니다.
	/=	a /= b	a에 b의 값을 나누고 대입합니다.
	&=	a &= b	a에 b의 값을 나눈 나머지를 대입합니다.

❖ 비교연산자와 대입연산자 실습을 위해 코드를 작성합니다.

- source/ch04/Comparison.java

결과

```
----------------------
비교연산자
a 〉 b는 true입니다.
a 〉= b는 true입니다.
a == b는 false입니다.
a != b는 true입니다.
대입연산자
r1 = a는 5입니다.
r2 += a는 15입니다.
r3 -= a는 5입니다.
r4 *= a는 50입니다.
r5 /= a는 2입니다.
r6 %= a는 0입니다.
```

❖ 논리연산자

[표 4-6] 논리연산자

	연산자	예시	의미
논리연산자	&&	a && b	a와 b가 모두 true일 경우 true를 반환합니다.
	&	a & b	a와 b가 모두 true일 경우 true를 반환합니다.
	||	a || b	a 또는 b가 true일 경우 true를 반환합니다.
	|	a | b	a 또는 b가 true일 경우 true를 반환합니다.
	!	a ! b	a와 반대의 값을 반환합니다.

❖ 논리연산자 실습을 위해 코드를 작성합니다.

- source/ch04/Logic.java

```
결과
---------------------
a && b는 false
a & b는 false
a || b는 true
a | b는 true
!a는 false
```

❖ 조건문 - if문의 구조

```
if(조건){조건이 true일 때 실행할 코드}
```

❖ if문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/If.java

```
결과
----------
점수는 100점입니다.
```

❖ 조건문 - if ~ else문의 구조

```
if(조건){조건이 true일 때 실행할 코드}else{조건이 false일 때 실행할 코드}
```

❖ if ~ else문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/Ifelse.java

```
결과
----------
점수는 100점이 아닙니다.
```

❖ 조건문 - else if문의 구조

```
if(조건){조건이 true일 때 실행할 코드}else if(조건2)
{조건2가 true일 때 실행할 코드}else if(조건3)
{조건3이 true일 때 실행할 코드}else if(조건4).....
```

❖ else if문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/Elseif.java

결과

점수는 100점이 아닙니다.

❖ 조건문 - switch문의 구조

```
switch (전달인자){
  case 조건1:
    조건1에 해당될 때 실행될 코드
      break;
   case 조건2:
    조건2에 해당될 때 실행될 코드
      break;
    default:
    모든 조건에 해당되지 않을 때 실행될 코드
   break;
```

❖ switch문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/Switch.java

결과

점수는 80점 입니다.

❖ 반복문 – for문의 구조

```
for (int i = 0; i < 반복횟수; i++) {반복할 코드}
```

❖ for문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/For.java

결과

```
1
2
3
4
5
6
7
8
9
10
a가 10이므로 for문은 종료됩니다.
```

❖ 반복문 – while문의 구조

```
while(조건){반복할 코드}
```

❖ while 문을 실습해보기 위해 다음의 코드를 작성합니다.

- source/ch04/While.java

결과

```
1
2
3
4
5
6
7
8
9
10
a가 10이므로 while문은 종료됩니다.
```

❖ break의 기능을 알아보기 위해 코드를 작성합니다.

- source/ch04/Break.java

결과

```
----------------------
1
2
3
4
5
```

❖ continue의 기능을 알아보기 위해 코드를 작성합니다.

- source/ch04/Continue.java

결과

```
----------
1
3
5
7
.....
47
49
```

클래스 및 객체

❖ 객체

- 메모리에 임시적으로 저장되는 프로그램의 구성요소

❖ 객체를 선언하고 생성하는 방법

```
클래스명 변수 = new 클래스명();
ex) Object o = new Object();
```

클래스 및 객체

❖ 클래스

- 객체를 만들기 위한 틀

클래스

객체

[그림 4-2] 클래스와 붕어빵틀

❖ 클래스의 구조

```
class 클래스이름 {}
```

❖ 상속

- 자식클래스가 부모클래스의 구성요소를 상속

❖ extends를 이용하여 클래스를 상속받는 방법

```
자식 클래스 명 extends 상속받을 부모 클래스 명
ex) public class Child extends Parents{}
```

❖ 상속을 구현하기 위해 부모 클래스(Parents)와 자식클래스(Child)를 작성

- source/ch04/Parents.java
- source/ch04/Child.java

결과

```
---------------------
부모클래스의 변수입니다.
```

클래스 및 객체

❖ 생성자

- 클래스가 메모리상에 객체를 생성시켰을 때 가장 먼저 수행할 작업을 기술하는 메소드

❖ 생성자의 구조

```
public 클래스와 동일한 이름{}
```

❖ 생성자를 이해하기 위한 코드를 작성

- source/ch04/Constructor.java

결과

```
-----------------
1
```

❖ 멤버변수
- 클래스에 속하며 클래스에 속한 모든 메소드에서 사용 가능

❖ 멤버변수를 이해하기 위한 코드를 작성
- source/ch04/Member.java

결과

```
---------------------------

지역변수입니다.
초기화하지 않은 멤버변수 : null
초기화한 멤버변수 : 멤버변수입니다.
```

❖ 접근제어자

- 클래스와 클래스 간 또는 패키지와 패키지 간의 접근 권한을 지정

[표 4-7] 접근제어자의 종류

접근 제어자	동일한 클래스	동일한 패키지	하위 클래스	전체
private	O			
default	O	O		
protected	O	O	O	
public	O	O	O	O

[표 4-8] 접근제어자의 사용대상

사용대상	사용 가능한 접근 제어자
클래스	public
멤버변수	public, protected, default, private
메소드	public, protected, default, private

❖ 접근제어자

- private – 동일한 클래스 내에서 접근 가능
  - source/ch04/Private.java * 주의 : 실행되지 않는 소스

결과

```
Exception in thread "main" java.lang.Error : 분석되지 않는 컴파일 문제점:
        OutSider1.s 필드가 가시적이지 않습니다.
        at ch04.Private.main(Private.java:10)
```

- protected – 동일한 클래스와 하위 패키지에서 접근 가능
  - source/ch04/Protected.java

결과

```
OutSider2
```

- public – 동일한 클래스, 패키지, 외부 패키지에서 접근 가능
  - source/ch04/Public.java

결과

```
부모클래스의 변수입니다.
```

❖ static

- 자동으로 객체 생성, 클래스 내부에서 공유
  - source/ch04/Static.java

```
결과
----------------------
static으로 선언된 변수입니다.
static으로 선언된 메소드입니다.
```

❖ final

- 클래스, 메소드, 변수 선언 시 사용

[표 4-9] final 예약어의 대상에 다른 의미

사용대상	final 예약어
클래스	하위 클래스에 대한 상속을 허용하지 않는다.
메소드	하위 클래스에서 static로 선언한 메소드를 오버라이드 할 수 없다.
변수	상수가 된다. 값을 변경할 수 없다.

❖ final

- 클래스를 final로 선언

```
final class 클래스명{}
ex) final class JSPStudy{}
```

- 메소드를 final로 선언

```
final 리턴타입 메소드명() {}
ex) final void jspStudy() {}
```

- 변수를 final로 선언

```
final 변수타입 변수명;
ex) final String jspStudy = "";
```

import 및 주석

❖ **import**

- 외부 패키지에 속한 클래스를 참조하기 위한 예약어

```
import 패키지명.클래스명;
ex) import out.Parents; // 'out' 패키지의 'Parents' 클래스를 참조
ex) import out.*; // 'out' 패키지에 속한 모든 클래스를 참조
```

❖ **import를 이해하기 위해 코드를 작성**

- source/ch04/Child.java

import 및 주석

❖ 주석

- 자바에서 제공하는 주석은 행 단위와 블록 단위 두 종류

[표 4-10] 주석

주석	단위
//내용	행
/*내용*/	블록

❖ 주석을 실습해보기 위한 코드를 작성

- source/ch04/Annotation.java

결과

```
----------
사과
키위
```

Thank You !