

A collection of various geometric shapes and patterns. The shapes include circles, squares, triangles, and polygons, some filled with solid colors like pink and others with patterns like dots or stripes. There are also abstract shapes like a cross and a wavy line. The shapes are scattered across the page, with some overlapping. The colors are primarily black, white, and pink. The patterns include dots, stripes, and wavy lines. The shapes are drawn with black outlines. The overall style is minimalist and geometric.

# Part1. 람다식

1) 람다식이란?

2) 기본 문법

3) 매개변수와 리턴값

- 매개변수 X / 리턴값 X
- 매개변수 0 / 리턴값 X
- 매개변수 0 / 리턴값 0

4) 클래스 멤버와 로컬 변수

5) 메소드 참조

- 정적 메소드 & 인스턴스 메소드 참조
- 매개 변수의 메소드 참조
- 생성자 참조

6) 연습문제

# 1) 람다식이란?

- 함수적 프로그래밍 > 병렬 처리와 이벤트 지향 프로그래밍에 적합
- 자바의 함수적 프로그래밍을 위해 자바8부터 람다식을 지원
- 익명 함수를 생성하기 위한 식
- 매개변수를 가지는 코드 블록의 형태로 나타나지만 = 함수를 정의하는 형태를 띠지만, 런타임 시에는 익명 구현 객체를 생성한다.

## 2) 기본 문법

**( 타입 매개변수, ... ) -> { 실행문, ... }**

ex. (int a, int b) -> { System.out.println(a+b); }

*1. 매개변수가 한 개인 경우*

(int a) -> { System.out.println(a); }

= (a) -> { System.out.println(a); }

= a -> { System.out.println(a); }

*2. 매개변수가 없을 경우*

( ) -> { 실행문, ... }

*3. 중괄호{ }에 return문만 있을 경우*

(a, b) -> { return a+b; }

= (a, b) -> a+b

*ex. 매개변수 1개에 return문만 있는 경우*

: a -> a+10

### 3) 매개변수와 리턴값

- 매개변수 X / 리턴값 X

```
MyFunctionalInterface.java ✖ MyFunctionalInterfaceExample.java
1 package week6_1;
2
3 @FunctionalInterface
4 public interface MyFunctionalInterface {
5     public void method();
6 }
```

```
MyFunctionalInterface.java MyFunctionalInterfaceExample.java ✖
1 package week6_1;
2
3 public class MyFunctionalInterfaceExample {
4     public static void main(String[] args) {
5         MyFunctionalInterface fi;
6
7         fi = () -> {
8             String str = "first call";
9             System.out.println(str);
10        };
11        fi.method();
12
13        fi = () -> { System.out.println("second call"); };
14        fi.method();
15
16        fi = () -> System.out.println("third call");
17        fi.method();
18    }
19 }
```

실행 결과

```
Console ✖
<terminated> M
first call
second call
third call
```

### 3) 매개변수와 리턴값

- 매개변수 0 / 리턴값 X

```
MyFunctionalInterface2.java  MyFunctionalInterfaceExample2.java
1 package week6_1;
2
3 @FunctionalInterface
4 public interface MyFunctionalInterface2 {
5     public void method(int x);
6 }
```

```
MyFunctionalInterface2.java  MyFunctionalInterfaceExample2.java
1 package week6_1;
2
3 public class MyFunctionalInterfaceExample2 {
4     public static void main(String[] args) {
5         MyFunctionalInterface2 fi;
6
7         fi = (x) -> {
8             int result = x * 3 + 2;
9             System.out.println(result);
10        };
11        fi.method(4);
12
13        fi = (x) -> { System.out.println(x * 3 + 2); };
14        fi.method(4);
15
16        fi = x -> System.out.println(x * 3 + 2);
17        fi.method(4);
18    }
19 }
```

실행 결과

```
Console
<terminated>
14
14
14
```

### 3) 매개변수와 리턴값

- 매개변수 0 / 리턴값 0

실행 결과

Console

<terminated>

63

63

63

63

MyFunctionalInterface3.java

MyFunctionalInterfaceExample3.java

```
1 package week6_1;
2
3 public class MyFunctionalInterfaceExample3 {
4     public static void main(String[] args) {
5         MyFunctionalInterface3 fi;
6
7         fi = (x, y) -> {
8             int result = x * y;
9             return result;
10        };
11        System.out.println(fi.method(7, 9));
12
13        fi = (x, y) -> { return x * y; };
14        System.out.println(fi.method(7, 9));
15
16        fi = (x, y) -> x * y;
17        System.out.println(fi.method(7, 9));
18
19        fi = (x, y) -> multiply(x, y);
20        System.out.println(fi.method(7, 9));
21    }
22
23    public static int multiply(int x, int y) {
24        return (x * y);
25    }
26 }
```

MyFunctionalInterface3.java

MyFunctionalInterfaceExample3.java

```
1 package week6_1;
2
3 @FunctionalInterface
4 public interface MyFunctionalInterface3 {
5     public int method(int x, int y);
6 }
```

## 4) 클래스 멤버와 로컬 변수

- 클래스 멤버

MyFunctionalInterface4.java UsingThis.java UsingThisExample.java

```
1 package week6_1;
2
3 public interface MyFunctionalInterface4 {
4     public void method();
5 }
```

MyFunctionalInterface4.java UsingThis.java UsingThisExample.java

```
1 package week6_1;
2
3 public class UsingThis {
4     public int outterField = 10;
5
6     class Inner {
7         int innerField = 20;
8
9         void method() {
10             //람다식
11             MyFunctionalInterface fi = () -> {
12                 System.out.println("outterField: " + outterField);
13                 System.out.println("outterField: " + UsingThis.this.outterField + "\n");
14
15                 System.out.println("innerField: " + innerField);
16                 System.out.println("innerField: " + this.innerField + "\n");
17             };
18             fi.method();
19         }
20     }
21 }
```

MyFunctionalInterface4.java UsingThis.java UsingThisExample.java

```
1 package week6_1;
2
3 public class UsingThisExample {
4     public static void main(String... args) {
5         UsingThis usingThis = new UsingThis();
6         UsingThis.Inner inner = usingThis.new Inner();
7         inner.method();
8     }
9 }
```

### 실행 결과

Console Problem

<terminated> UsingThi

outterField: 10  
outterField: 10

innerField: 20  
innerField: 20



## 4) 클래스 멤버와 로컬 변수

- 로컬 변수

```
MyFunctionalInterface5.java UsingLocalVariable.java
1 package week6_1;
2
3 public interface MyFunctionalInterface5 {
4     public void method();
5 }
```

```
MyFunctionalInterface5.java UsingLocalVariable.java
1 package week6_1;
2
3 public class UsingLocalVariable {
4     void method(int arg) { //arg는 final 특성을 가짐
5         int localVar = 40; //localVar는 final 특성을 가짐
6
7         //arg = 31; //final 특성 때문에 수정 불가
8         //localVar = 41; //final 특성 때문에 수정 불가
9
10        //람다식
11        MyFunctionalInterface fi = () -> {
12            //로컬변수 사용
13            System.out.println("arg: " + arg);
14            System.out.println("localVar: " + localVar + "\n");
15        };
16        fi.method();
17    }
18 }
```

```
UsingLocalVariableExample.java UsingLocalVariable.java
1 package week6_1;
2
3 public class UsingLocalVariableExample {
4     public static void main(String... args) {
5         UsingLocalVariable ulv = new UsingLocalVariable();
6         ulv.method(20);
7     }
8 }
```

실행 결과

```
Console Pr
<terminated> Usin
arg: 20
localVar: 40
```

## 5) 메소드 참조

- 정적 메소드 & 인스턴스 메소드 참조

```
Calculator.java ✕ methodEx.java
1 package week6_1;
2
3 public class Calculator {
4
5     //정적 메소드
6     public static int staticMethod(int a, int b) {
7         return a+b;
8     }
9     //인스턴스 메소드
10    public static int instanceMethod(int a, int b) {
11        return a+b;
12    }
13 }
```

## 5) 메소드 참조

- 정적 메소드 & 인스턴스 메소드 참조

```
Calculator.java *methodEx.java ✖
1 package week6_1;
2 import java.util.function.IntBinaryOperator;
3
4 public class methodEx {
5     public static void main(String[] args) {
6         IntBinaryOperator opt;
7
8         //정적 메소드 참조
9         opt = (x,y) -> Calculator.staticMethod(x,y); //람다식
10        System.out.println("result1 : " + opt.applyAsInt(10, 40));
11
12        opt = Calculator::staticMethod; //메소드 참조
13        System.out.println("result2 : " + opt.applyAsInt(20, 30));
14
15        //인스턴스 메소드 참조
16        Calculator obt = new Calculator();
17        opt = (x,y) -> obt.instanceMethod(x,y); //람다식
18        System.out.println("result3 : " + opt.applyAsInt(30, 70));
19
20        opt = Calculator::instanceMethod; //메소드 참조
21        System.out.println("result4 : " + opt.applyAsInt(20, 80));
22    }
23 }
```

실행 결과

```
Problems @ Javadoc De
<terminated> methodEx [Java
result1 : 50
result2 : 50
result3 : 100
result4 : 100
```

## 5) 메소드 참조

- 매개 변수의 메소드 참조

```
1 package week6_1;
2 import java.util.function.ToIntBiFunction;
3
4 public class argumentMethodEx {
5     public static void main(String[] args) {
6         ToIntBiFunction<String,String> func;
7
8         func = (a,b) -> a.compareToIgnoreCase(b); //람다식
9         println(func.applyAsInt("아미공", "아미공"));
10
11         func = String::compareToIgnoreCase; //메소드 참조
12         println(func.applyAsInt("고급자바", "고급자바"));
13     }
14
15     //println() 메소드 정의, 기존의 println 메소드는 argumentMethodEx 객체 형식을 출력할 수 없기 때문
16     private static void println(int order) {
17         if(order<0) {System.out.println("in dictionary order.");}
18         else if(order==0) {System.out.println("same string.");}
19         else {System.out.println("in reverse order.");}
20     }
21 }
```

실행 결과

Problems @ Javadoc Declarati  
<terminated> argumentMethodEx [Ja  
same string.  
same string.

# 5) 메소드 참조

- 생성자 참조

```
ConstructorEx.java member.java
1 package week6_1;
2
3 public class member {
4
5     private String name;
6     private String id;
7
8     public member() {
9         System.out.println("start member()");
10    }
11    public member(String id) {
12        System.out.println("start member(id)");
13    }
14    public member(String name, String id) {
15        System.out.println("start member(name, id)");
16        this.name = name;
17        this.id = id;
18    }
19
20    public String getId() { return id;}
21 }
```

```
ConstructorEx.java member.java
1 package week6_1;
2 import java.lang.reflect.Member;
3 import java.util.function.*;
4
5 public class ConstructorEx {
6     public static void main(String[] args) {
7         Function<String, member> func1 = member::new; //생성자 참조
8         member m1 = func1.apply("amigong");
9
10        BiFunction<String, String, member> func2 = member::new; //생성자 참조
11        member m2 = func2.apply("amigong", "아미공");
12    }
13 }
```

실행 결과

```
Problems @ Javadoc Declara
<terminated> ConstructorEx [Java A]
start member(id)
start member(name, id)
```

## 6) 연습문제

```
*LambdaEx.java
1 package week6_1;
2 import java.util.function.*;
3
4 public class LambdaEx {
5     public static int method(int a, int b) {
6         IntSupplier s = () -> {
7             a *= 100; //지역변수 값 변경 불가능
8             int result = a+b;
9             return result;
10        };
11        int result = s.getAsInt();
12        return result;
13    }
14 }
```

# 출석 과제 (5/3 월 오후 11:55 마감)

Q. 잘못 작성된 람다식을 고르시오.

- ①  $a \rightarrow a+3$
- ②  $a, b \rightarrow a * b$
- ③  $x \rightarrow \text{System.out.println}(x/5)$

위의 기간까지 고급 자바 실습  
DS-CLASS 에 제출하시기 바랍니다.