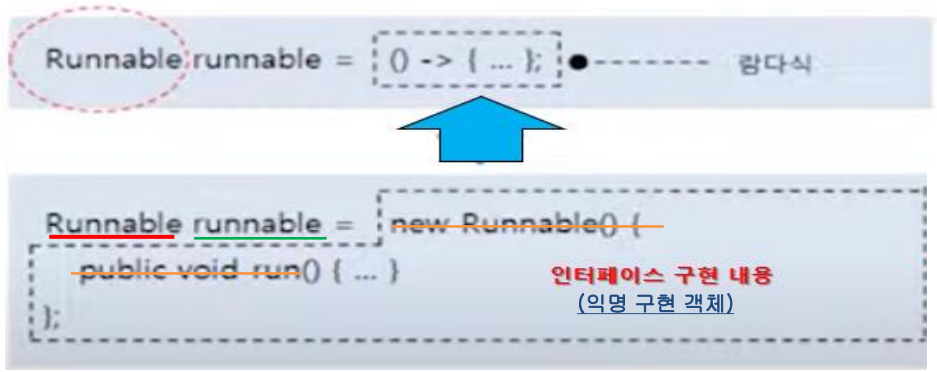
람다식이란

▣ 람다식이란

- 자바는 함수형 프로그래밍을 위해 Java 8부터 람다식(Lambda Expressions)을 지원한다.
- 자바는 람다식을 익명 구현 객체로 변환한다.
- 인터페이스의 익명 구현 객체를 람다식으로 표현하려면 인터페이스가 단 하나의 추상 메소드만 가져야 한다.
- 인터페이스가 단 하나의 추상 메소드를 가질 때, 이를 함수형 인터페이스(functional interface)라고 한다.
- 인터페이스가 함수형 인터페이스임을 보장하기 위해서는 @FunctionalInterface 어노테이션을 붙이면 된다.(필수x)

- 어떤 인터페이스를 구현할지는 대입되는 인터페이스에 달려있다.(★)




```
public interface Calculable {
  //추상 메소드
  void calculate(int x, int y);
}
```

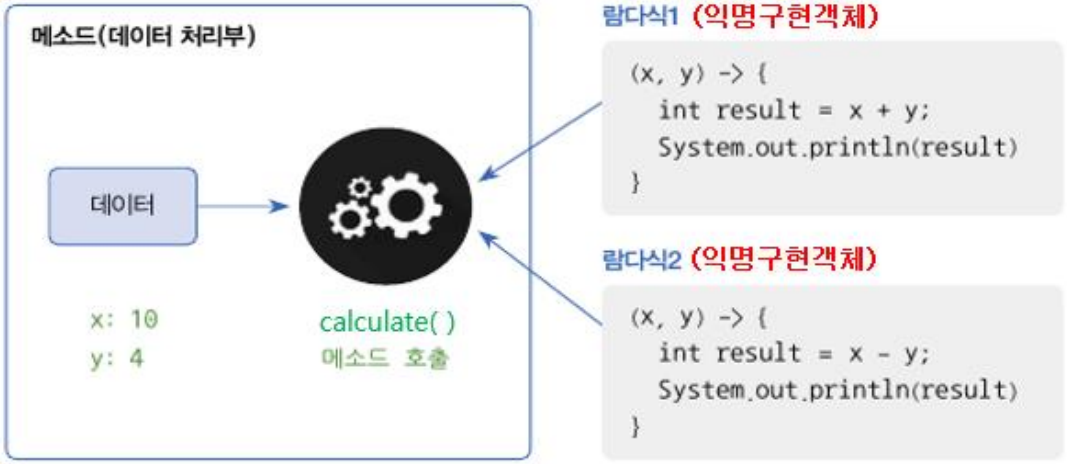
```
public void action(Calculable calculable) {
  int x = 10;
  int y = 4;
  calculable.calculate(x, y);    //데이터를 제공하고 추상 메소드를 호출
}
```

```
action( (x, y) -> {
  int result = x + y;
  System.out.println(result);
});
```

(익명 구현 객체)

※ 메소드는 class, object 안에만 존재하나, 함수는 단독으로 존재하며 실행코드를 갖고 있는 블럭이다.

- 데이터 처리부는 데이터만 가지고 있을 뿐 -> 처리 방법이 정해져 있지 않아 외부에서 제공된 함수에 의존한다.
- 데이터 처리부는 람다식을 받아 -> 매개변수에 데이터를 대입하고 -> 함수의 중괄호를 실행시켜 처리한다.




```
public void action(Calculable calculable) {   //메소드(데이터 처리부)
  int x = 10;  //데이터
  int y = 4;   //데이터
  calculable.calculate(x, y);    //데이터를 제공하고 추상 메소드를 호출
}              //메소드 호출
```

- 데이터 처리부는 제공된 함수의 입력값으로 데이터를 넣고 함수에 정의된 처리 내용을 실행한다.
- 동일한 데이터라도 함수A를 제공해서 처리하는 결과와 함수B를 제공해서 처리하는 결과는 다를 수 있다.
- 이것이 함수형 프로그래밍의 특징으로, 데이터 처리의 다형성이라고 볼 수 있다.