## 一、函数式接口

**一、 什么是函数式接口**

    函数式接口(Functional Interface)就是一个有且仅有一个抽象方法，但是可以有多个非抽象方法的接口。

**二、 函数式接口作用是什么？**

适用场景：主要用于函数式编程（即使用lambda表达式编程），函数式接口可以使用于lambda进行参数定义。

**三、使用流程**

1）定义函数式接口

2）定义方法使用上面定义的接口进行参数接收，并调用接口中的方法进行逻辑处理

3）具体使用2中的方法的时候，传递lambda表达式进行实际上的逻辑处理

实例：

1. 函数式接口

/\*\*

\* @FunctionalInterface 注解可以检测接口是否为函数式接口

\* 如果不是函数式接口，则编译失败

\* @param <T>\*/

@FunctionalInterfacepublic interface Compute<T> {

public int add(int a , int b );// public int add(int a , int b );

}

1. 函数式接口调用

-1）匿名函数方式

- 2）lambda方式

public class Test1 {

public static int add(int a,int b, Compute<Integer> compute ){// compute.add()

return compute.add(a,b) ;

}

public static void main(String[] args) {

test2() ;

}

//传统匿名类方式

public static void test2( ) {

int i = 1 ;

int j =2 ;

add( i , j , new Compute<Integer>(){

@Override

public int add(int a, int b) {

return a+b;

}

});

}

//使用过lamba表达式进行参数调用处理

public static void test1() {

int i = 1 ;

int j =2 ;

add( i , j ,(a,b)->{

return a+b ;

});

}

}

好处：使用lambda方式使调用更加的简洁

个人理解：

1）在函数式接口的调用的时候，如果传递了lambda则函数式接口内部调用了lambda表达式，最终使数据得以处理

2）lambda相当于一个函数，有输入和输出，当然输入和输出也可以是空的

3）那么函数式接口在什么时候使用呢？

    当我们需要封装一段逻辑，需要使用lamdba处理的时候，那么我们就创建一个函数式接口作为参数，这个时候就可以传递lambda来处理逻辑了

## java提供的内置函数式接口及其具体使用详解

1. java.util.function.Supplier<T> 接口包含一个无参方法：T get()

 主要作用：接收一个lambda执行业务逻辑，返回对应的数据类型T（无输入，只有指定类型的输出）

    例子：

[IMG_256](javascript:void(0);)public static String test3(Supplier<String> supplier ){

return supplier.get() ;

}

public static void main(String[] args) {

String str = test3(()->{

return "测试supplier接口" ;

}) ;

System.out.println( str );

}

1. java.util.function.Consumer<T> 消费一个数据，数据类型又泛型决定

主要作用：接收一个lambda执行业务逻辑，输入的数据类型为T（有输入，无输出）

[IMG_256](javascript:void(0);)

public static void test3(String t , Consumer<String> consumer ){

consumer.accept(t); ;

}

public static void main(String[] args) {

test3("测试",(str)->{

System.out.println( str );

}) ;

}

1. java.util.function.Predicate<T>接口

作用：接收一个lambda执行业务逻辑，对某种数据类型的数据（传入的参数）进行判断，结果返回一个boolean值（）

例子：

[IMG_256](javascript:void(0);)

//对传入的参数t进行逻辑判断，返回true或者false

public static boolean test3(int t , Predicate<Integer> predicate ){

return predicate.test(t);

}

public static void main(String[] args) {

test3(10 ,(i)->{

if(i>5){

return true;

}else{

return false;

}

}) ;

}

4）java.util.function.Function<T,R>接口用来根据一个类型的数据得到另一个类型的数据

作用：接收一个lambda执行业务逻辑，对某种数据类型的数据进行处理，返回固定的数据类型（有输入，有输出）

例子：

[IMG_256](javascript:void(0);)

public static Integer test4(String str, Function<String,Integer> function ){

return function.apply(str) ;

}

public static void main(String[] args) {

int k = test4("10" ,(str)->{

return Integer.parseInt(str) ;

}) ;

System.out.println( k );

}