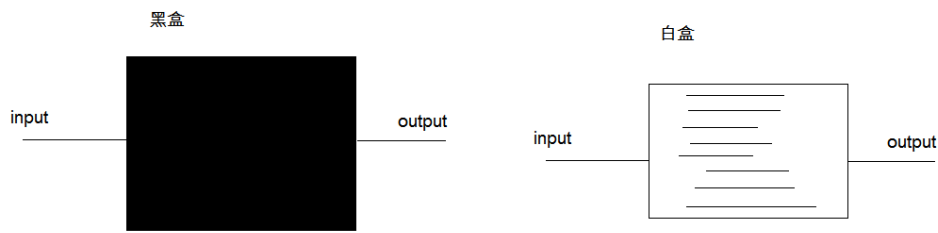


Junit单元测试:



* 测试分类:

1. 黑盒测试: 不需要写代码, 给输入值, 看程序是否能够输出期望的值。
2. 白盒测试: 需要写代码的。关注程序具体的执行流程。

* Junit使用: 白盒测试

* 步骤:

1. 定义一个测试类(测试用例)

* 建议:

- * 测试类名: 被测试的类名Test calculatorTest
- * 包名: xxx.xxx.xx.test cn.itcast.test

2. 定义测试方法: 可以独立运行

* 建议:

- * 方法名: test测试的方法名 testAdd()
- * 返回值: void
- * 参数列表: 空参

3. 给方法加@Test

4. 导入junit依赖环境

* 判定结果:

- * 红色: 失败
- * 绿色: 成功

* 一般我们会使用断言操作来处理结果 (Assert是一个类, assertEquals是一个静态方法)

* Assert.assertEquals(期望的结果, 运算的结果);

* 补充:

* @Before:

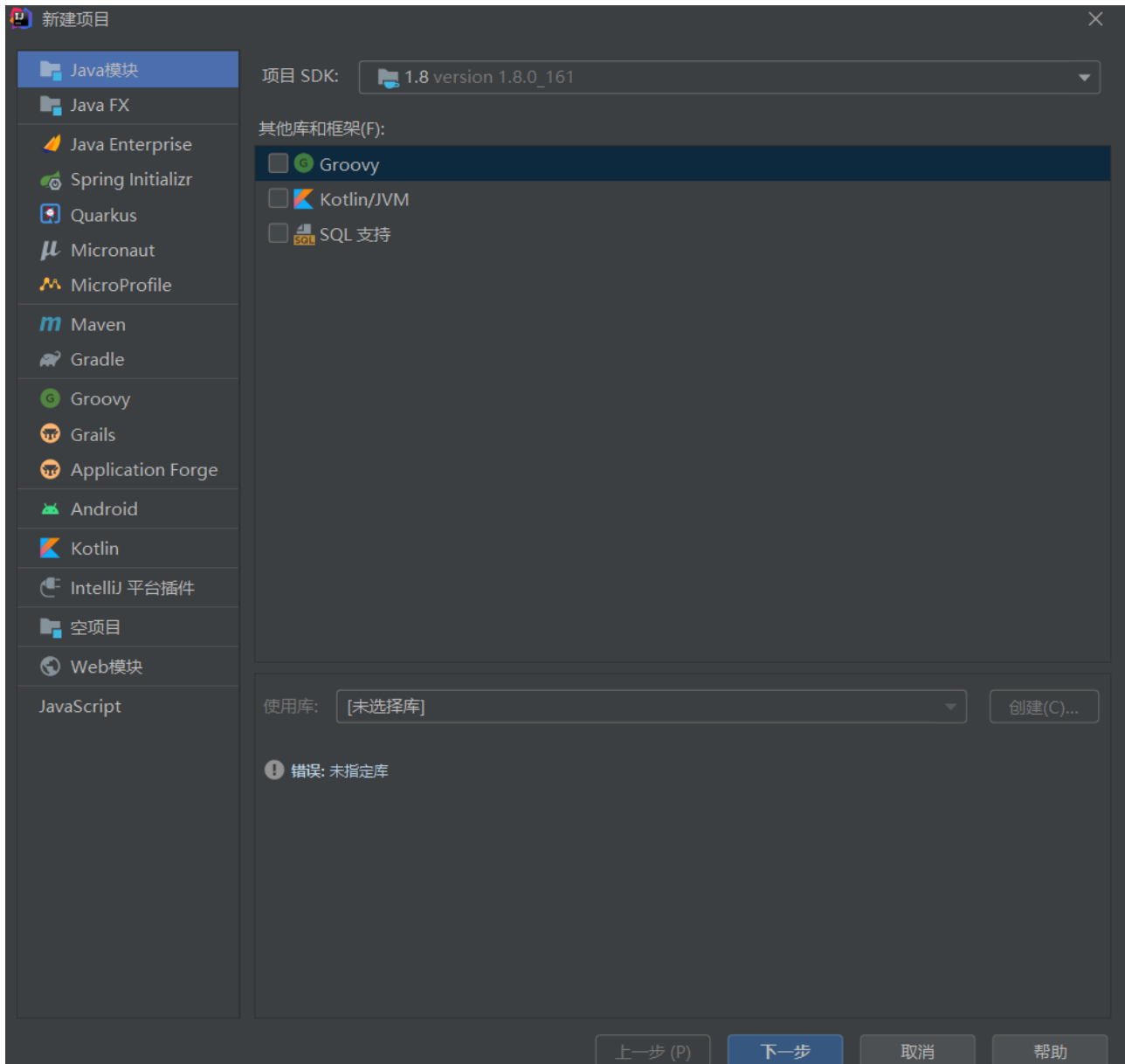
* 修饰的方法会在测试方法之前被自动执行

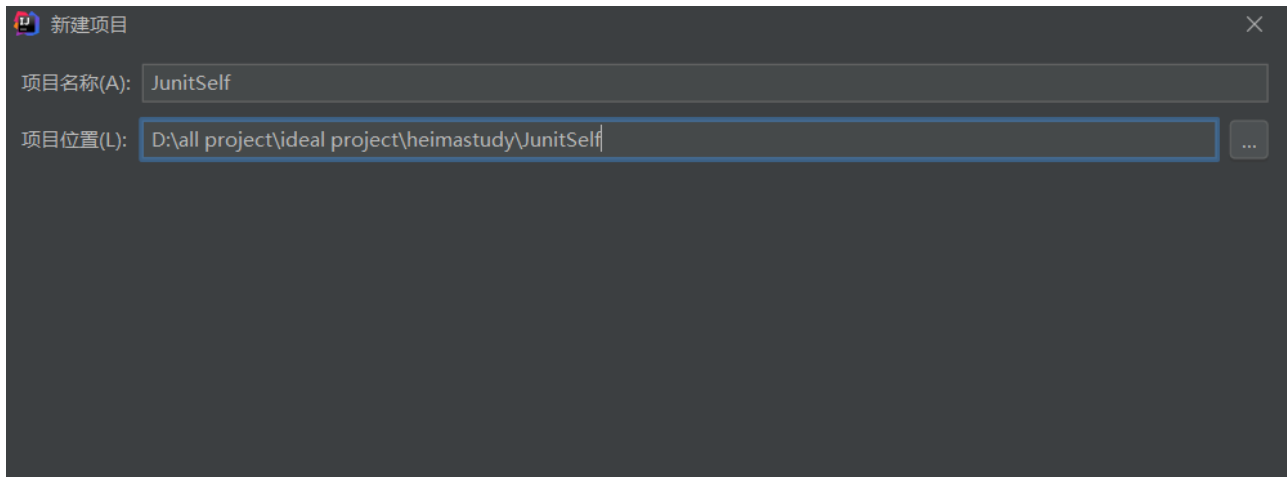
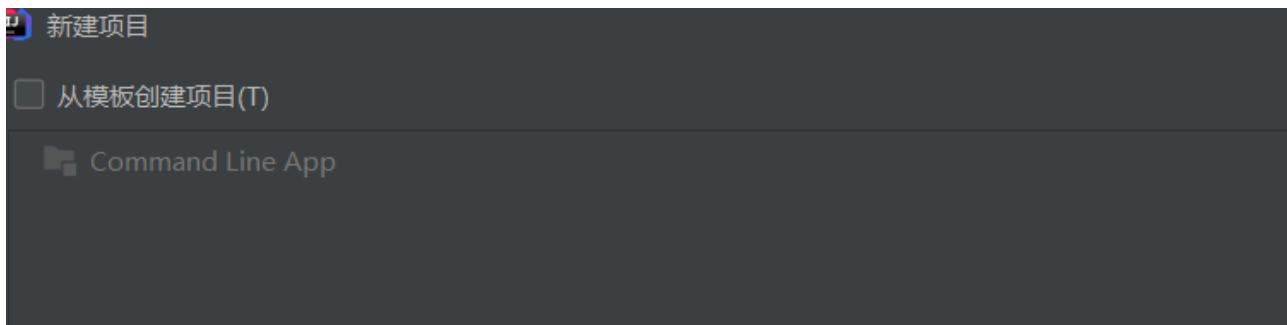
* @After:

* 修饰的方法会在测试方法执行之后自动被执行

Junit中IDEA应用

- 新建步骤





- 剩余在IDEA里看自己写的

JUnit输出查看

- 单独成功运行@Test

```
4  
5     public int add (int a , int b){  
6         //int i = 3/0;  
7  
8         //return a - b;  
9         return a+b;  
10    }
```

```

/**
 * 测试add方法
 */
@Test//加上@Test后即使不需要main也能独立运行'
    /**写上后发现会爆红，则需要手动到本地仓库里复制一个JUnit
    然后运行test发现还是会爆红，需要D:\CodeSoftware\Mvn
    */
    //运行时点击旁边的绿色三角形---再点击'testAdd()'，可以
    public void testAdd(){
        //1.创建计算器对象
        System.out.println("testAdd...");
        Calculator c = new Calculator();

        //2.调用add方法
        int result = c.add( a: 1, b: 2);
        //System.out.println(result);不用这个测试而是用断言
        //3.断言 我断言这个结果是3
        Assert.assertEquals( expected: 3,result);
    }

```

```

» ✓ 测试 已通过: 1 共 1 个测试 - 16毫秒
毫秒 D:\jdk-8u161-windows-x64\bin\java.exe ...
毫秒 testAdd...

进程已结束，退出代码为 0
|

```

- 单独失败运行@Test

```

    public int add (int a , int b){
        //int i = 3/0;

        return a - b;
    }

```

@Test//加上@Test后即使不需要main也能独立运行'

/*写上后发现会爆红，则需要手动到本地仓库里复制一个Junit-j
然后运行test发现还是会爆红，需要D:\CodeSoftware\MvnRe
*/

//运行时点击旁边的绿色三角形---再点击'testAdd()'，可以单

```
public void testAdd(){
```

//1.创建计算器对象

```
System.out.println("testAdd...");
```

```
Calculator c = new Calculator();
```

//2.调用add方法

```
int result = c.add( a: 1, b: 2);
```

//System.out.println(result);不用这个测试而是用断言进

//3.断言 我断言这个结果是3

```
Assert.assertEquals( expected: 3,result);
```

```
}
```

```
>> ❌ 测试 失败: 1 共 1 个测试 - 25毫秒

5毫秒 D:\jdk-8u161-windows-x64\bin\java.exe ...
5毫秒 testAdd...

java.lang.AssertionError:
预期: 3
实际: -1
<点击以查看差异>

<1 个内部行>
at org.junit.Assert.failNotEquals(Assert.java:835) <2 个内部行>
at cn.itcast.test.CalculatorTest.testAdd(CalculatorTest.java:46) <25 个内部行>

进程已结束，退出代码为 -1
```

- 成功运行@Test、@Before、@Close

```
public int add (int a , int b){  
    //int i = 3/0;  
  
    //return a - b;  
    return a+b;  
}
```

```

@Before
public void init(){
    System.out.println("init...");
}

/**
 * 释放资源方法:
 * 在所有测试方法执行完后, 都会自动执行该方法
 */
@After
public void close(){
    System.out.println("close...");
}

/**
 * 测试add方法
 */
@Test//加上@Test后即使不需要main也能独立运行'
    /*写上后会爆红, 则需要手动到本地仓库里复制一个Junit-jar包 -
    然后运行test发现还是会爆红, 需要D:\CodeSoftware\MvnReposit
    */
    //运行时点击旁边的绿色三角形---再点击'testAdd()', 可以单独运行
    public void testAdd(){
        //1.创建计算器对象
        System.out.println("testAdd...");
        Calculator c = new Calculator();
        //2.调用add方法
        int result = c.add( a: 1, b: 2);
        //System.out.println(result);不用这个测试而是用断言进行测试
        //3.断言 我断言这个结果是3
        Assert.assertEquals( expected: 3,result);
    }

```

```
D:\jdk-8u161-windows-x64\bin\java.exe ...  
init...  
testAdd...  
close...  
  
进程已结束，退出代码为 0
```

- 失败运行@Test、@Before、@After

```
public int add (int a , int b){  
    //int i = 3/0;  
    return a - b;  
    //return a+b;  
}
```



```
13      */
14      @Before
15      public void init(){
16          System.out.println("init...");
17      }
18      /**
19       * 释放资源方法:
20       * 在所有测试方法执行完后, 都会自动执行该方法
21       */
22      @After
23      public void close(){
24          System.out.println("close...");
25      }
26      /**
27       * 测试add方法
28       */
29      @Test//加上@Test后即使不需要main也能独立运行'
30          /*写上后发现会爆红, 则需要手动到本地仓库里复制一个JUnit-jar包
31          然后运行test发现还是会爆红, 需要D:\CodeSoftware\MvnRepos
32          */
33          //运行时点击旁边的绿色三角形--再点击'testAdd()', 可以单独运
34      public void testAdd(){
35          //1. 创建计算器对象
36          System.out.println("testAdd...");
37          Calculator c = new Calculator();
38          //2. 调用add方法
39          int result = c.add( a: 1, b: 2);
40          //System.out.println(result);不用这个测试而是用断言进行测试
41          //3. 断言 我断言这个结果是3
42          Assert.assertEquals( expected: 3, result); }
```

>> ❌ 测试失败: 1共 1 个测试 - 32毫秒

秒 D:\jdk-8u161-windows-x64\bin\java.exe ...

秒 init...

testAdd...

close...

java.lang.AssertionError:

预期:3

实际:-1

[<点击以查看差异>](#)

田 <1 个内部行>

田 at org.junit.Assert.failNotEquals(Assert.java:835) <2 个内部行>

田 at cn.itcast.test.CalculatorTest.testAdd(CalculatorTest.java:42) <27 个内部行>

进程已结束，退出代码为 -1