1. Junit单元测试
2. 反射
3. 注解
4. 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(测试用类)

建议：

测试类名：被测试类类名Test CalculatorTest

包名：xxx.xxx.xx.test cn.incast.test

1. 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

1. 给方法加@Test
2. 导入junit的依赖环境

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.assertEquals(3,result);

补充：

@Before

修饰的方法会在测试方法之前被执行

@After

修饰的方法会在测试方法之后被执行

2.反射：框架设计的灵魂

框架：半成品软件，可以在框架的基础上进行软件的开发，简化代码

反射：将类的各个部分封装为其他对象，这就是反射机制

好处：

1. 可以在程序的运行过程中，操作这些对象。
2. 可以解耦，提高程序的可扩展性。

获取Class对象的三种方式：

1. Class.forName(“全类名”):将字节码文件加载进内存，返回Class对象。（多用于配置文件，将类名定义在配置文件中，读取文件，加载类）
2. 类名.class：通过类名的属性来获取。（多用于参数的传递）
3. 对象.getClass():getClass()方法在Object类中定义。（多用于对象的获取字节码的方式）

结论：

同一个字节码文件(\*.class)再一次程序运行过程中，只会被加载一次，不论通过一种方式获取的对象都是同一个。

Class对象功能：

获取功能：

1. 获取成员变量们
   1. Field[] getFields():是来获取所有public修饰的成员变量的。
   2. Field getField(String name)：是来指定名称的public修饰的成员变量的。
   3. Field[] getDeclaredFields():获取所有的成员变量，不考虑修饰符
   4. Field getDeclaredField(String name):
2. 获取构造方法们
   1. Constructor[] getConstructors()
   2. Constructor<T>getConstructor(Class...parameterTypes)
   3. Constructor<T>getDeclaredConstructor(Class...parameterTypes)
   4. Constructor[] getDeclaredConstructors()
3. 获取成员方法们
   1. Method[] getMethods()
   2. Method getMethod(String name, Class... parameterTypes)
   3. Method[] getDeclaredMethods()
   4. Method getDeclaredMethod(String name, Class... parameterTypes)
4. 获取类名
   1. String getName()

Field:成员变量

操作：

1. 设置值

Void set(Object obj,Object value)

1. 获取值

get(Object obj)

1. 忽略访问权限修饰符的安全检查

setAccessible(true):暴力反射

Constructor:构造方法

创建对象：

T newInstance(Object... initargs)

如果构造使用空参数的方法创建对象，操作可以简化：Class对象的newInstance

Method:方法对象

执行方法：

Object invoke(Object obj,Object... args)

获取方法名称：

String getName:获取方法名

案例：

需求：写一个“框架”，在不能被改编任意代码的前提下，可以帮我们去创建任意类的对象，并且去执行任意的方法。

实现：

1. 配置文件
2. 反射

步骤：

1. 将需要创建的对象的全类名和需要执行的方法定义在配置文件中
2. 在程序加载的过程中配置文件
3. 使用反射技术来加载文件进内存
4. 创建对象
5. 执行方法