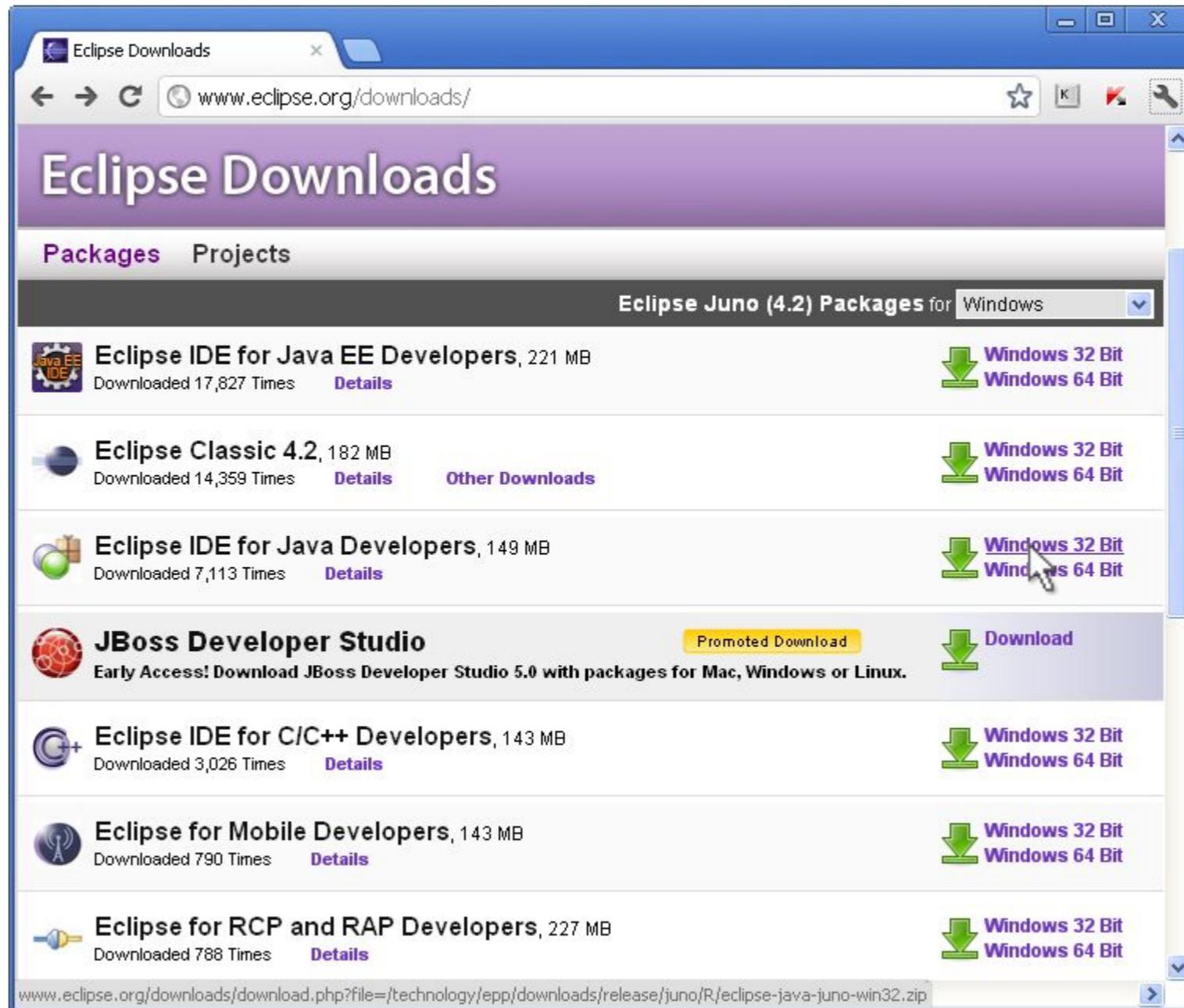




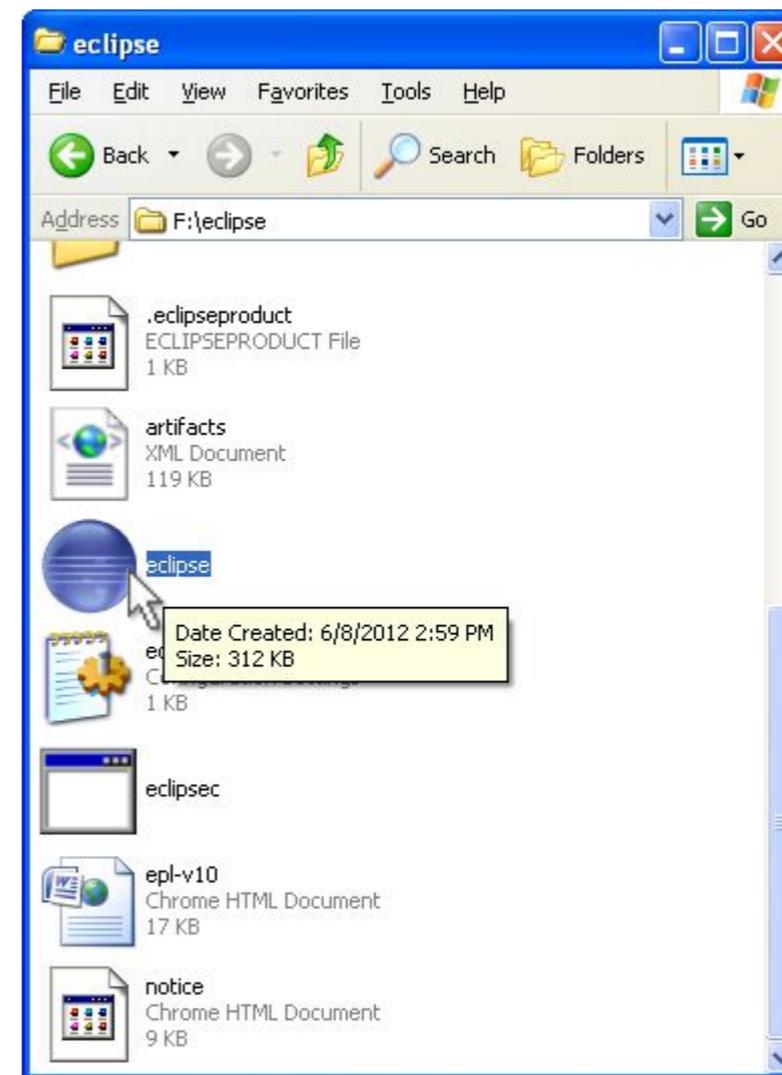
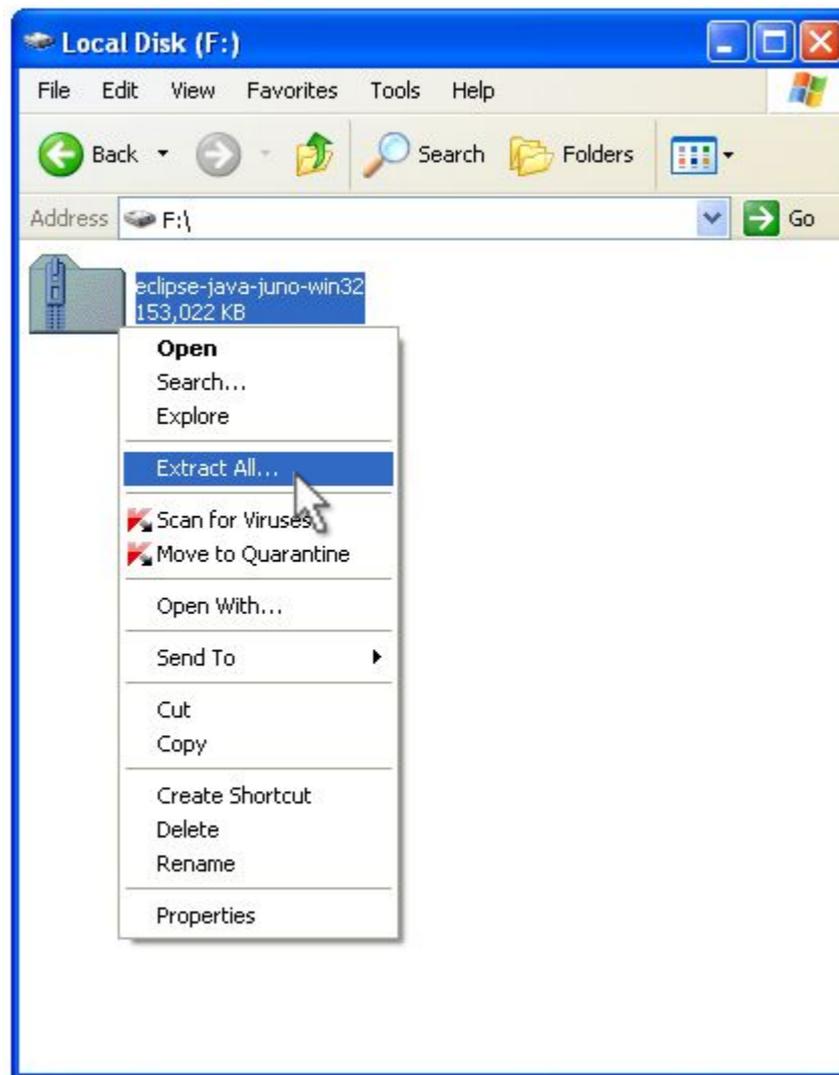
1. Введение

5. Среда разработки Eclipse

Установка Eclipse



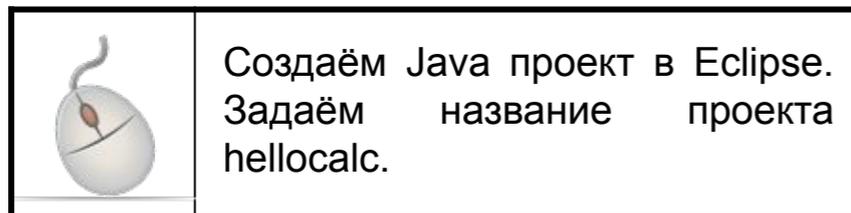
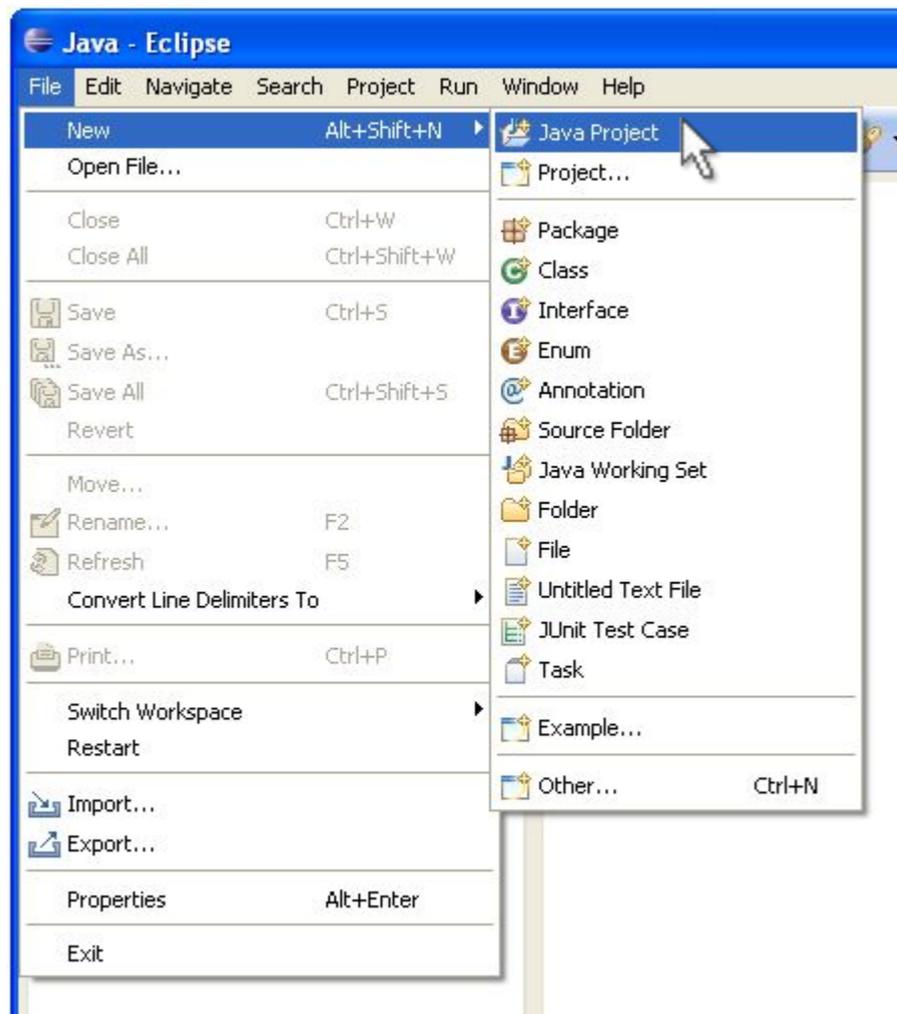
Eclipse можно загрузить с сайта www.eclipse.org/downloads. В данном курсе мы будем использовать Eclipse IDE for Java Developers.



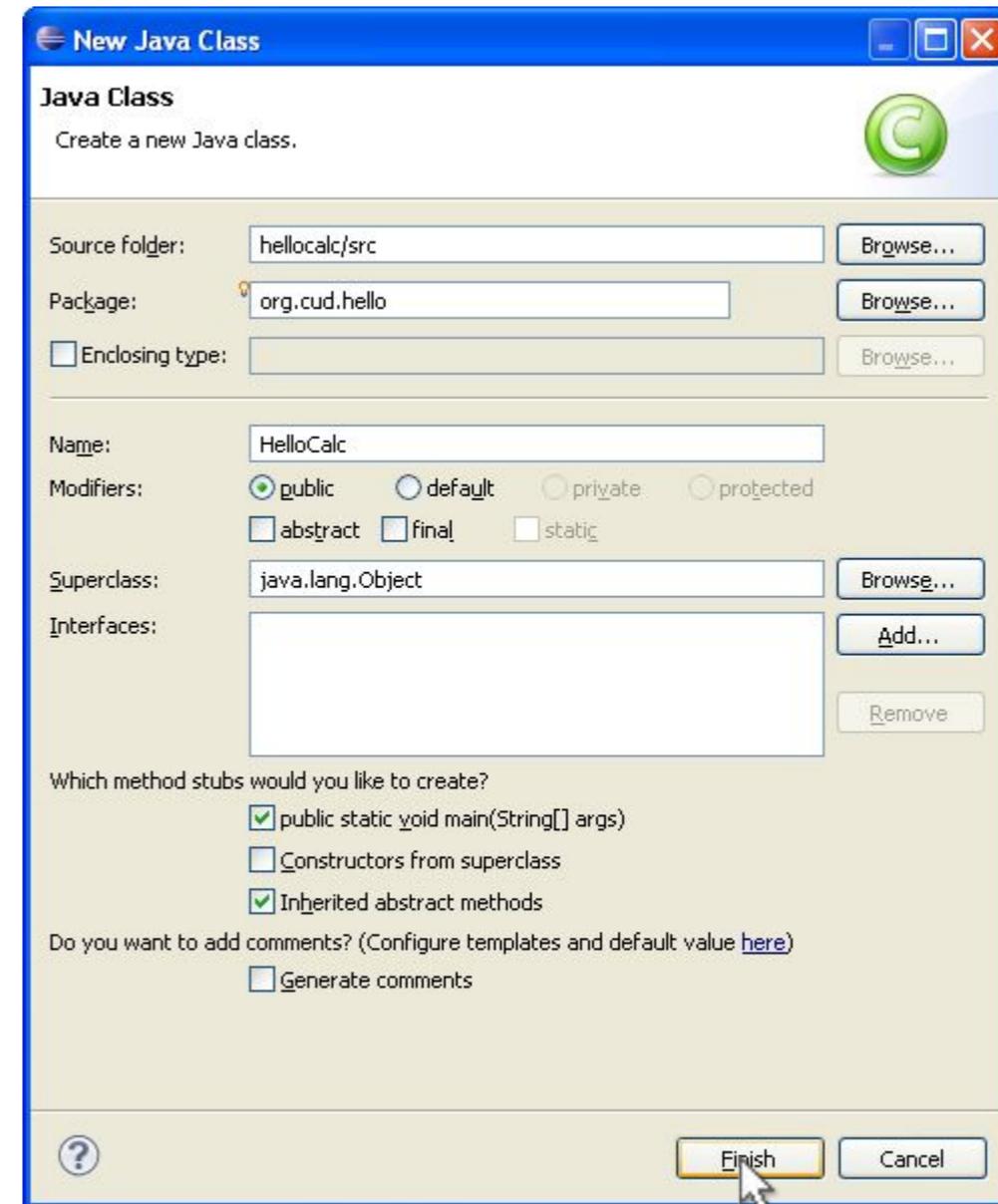
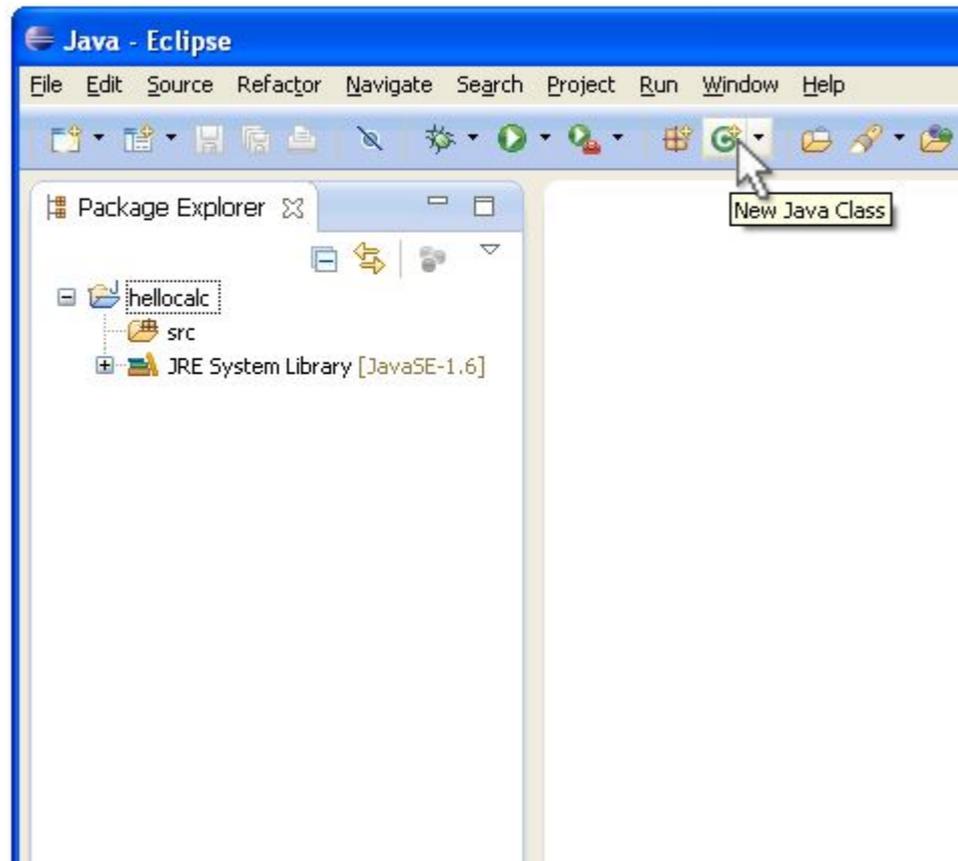
 Для установки Eclipse достаточно разархивировать загруженный zip файл.

Создание и запуск приложения

Создание проекта в Eclipse

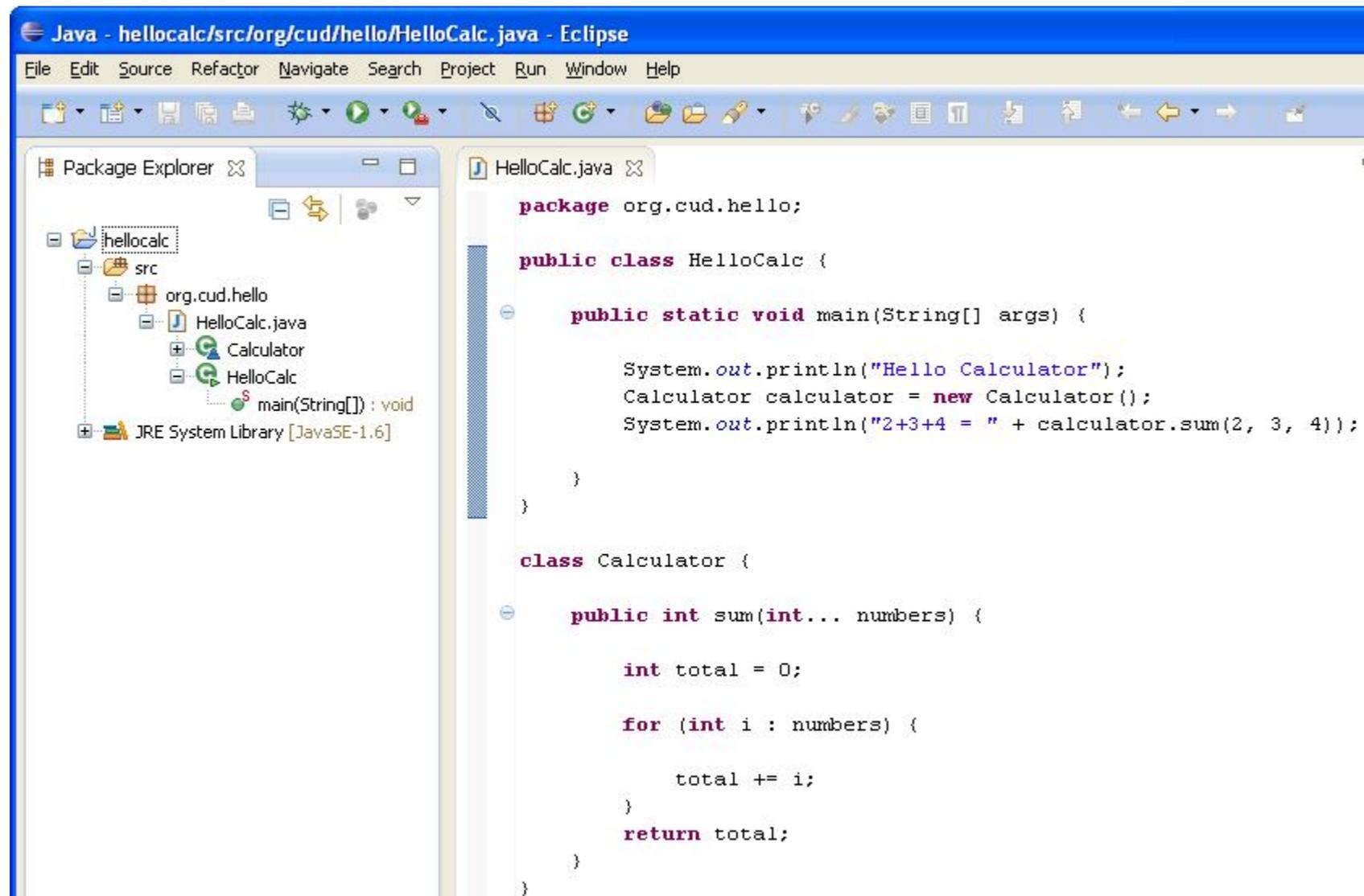


Создание класса



	<p>Создаём класс HelloCalc в пакете org.cud.hello со статическим методом main. Этот класс будет точкой входа приложения.</p>
---	--

Внесение изменений и создание другого класса



```
Java - hellocalc/src/org/cud/hello/HelloCalc.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer
hellocalc
src
  org.cud.hello
    HelloCalc.java
    Calculator
    HelloCalc
    main(String[]): void
  JRE System Library [JavaSE-1.6]

HelloCalc.java
package org.cud.hello;

public class HelloCalc {

    public static void main(String[] args) {

        System.out.println("Hello Calculator");
        Calculator calculator = new Calculator();
        System.out.println("2+3+4 = " + calculator.sum(2, 3, 4));

    }

}

class Calculator {

    public int sum(int... numbers) {

        int total = 0;

        for (int i : numbers) {

            total += i;

        }

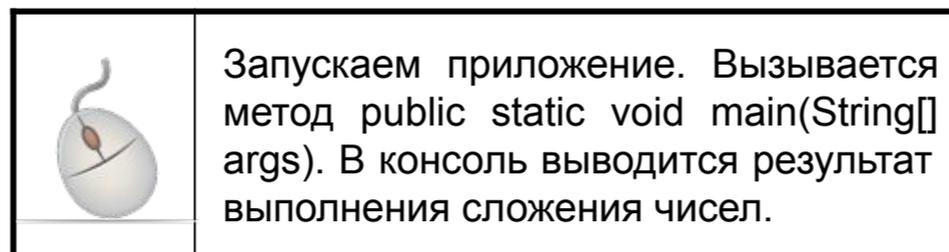
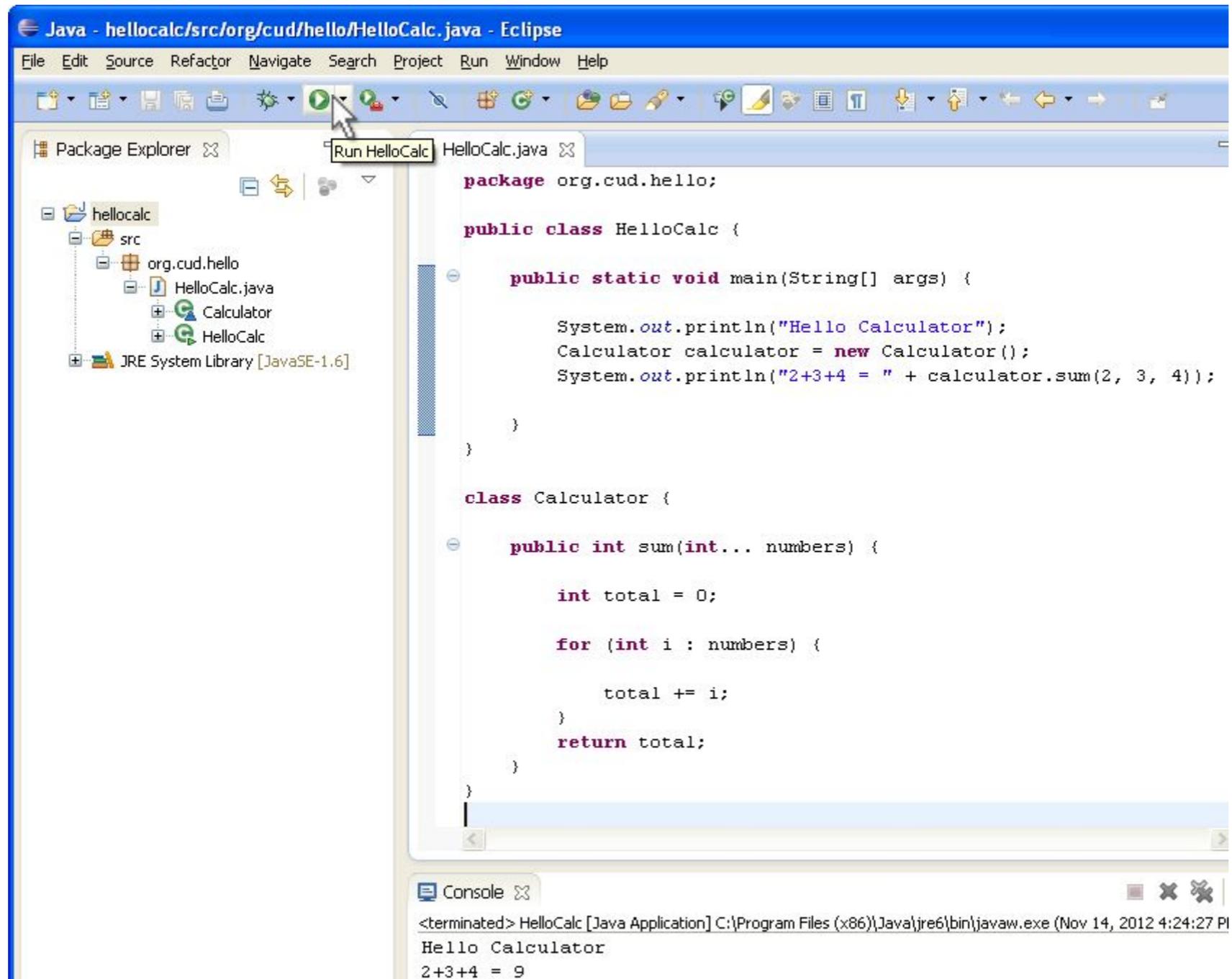
        return total;

    }

}
```

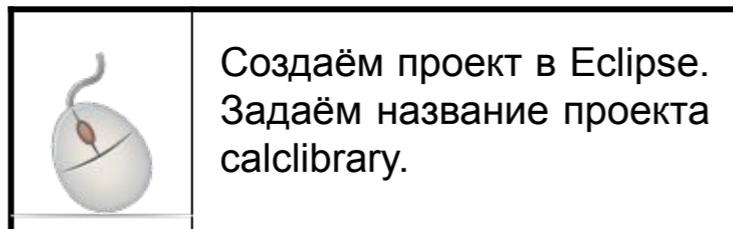
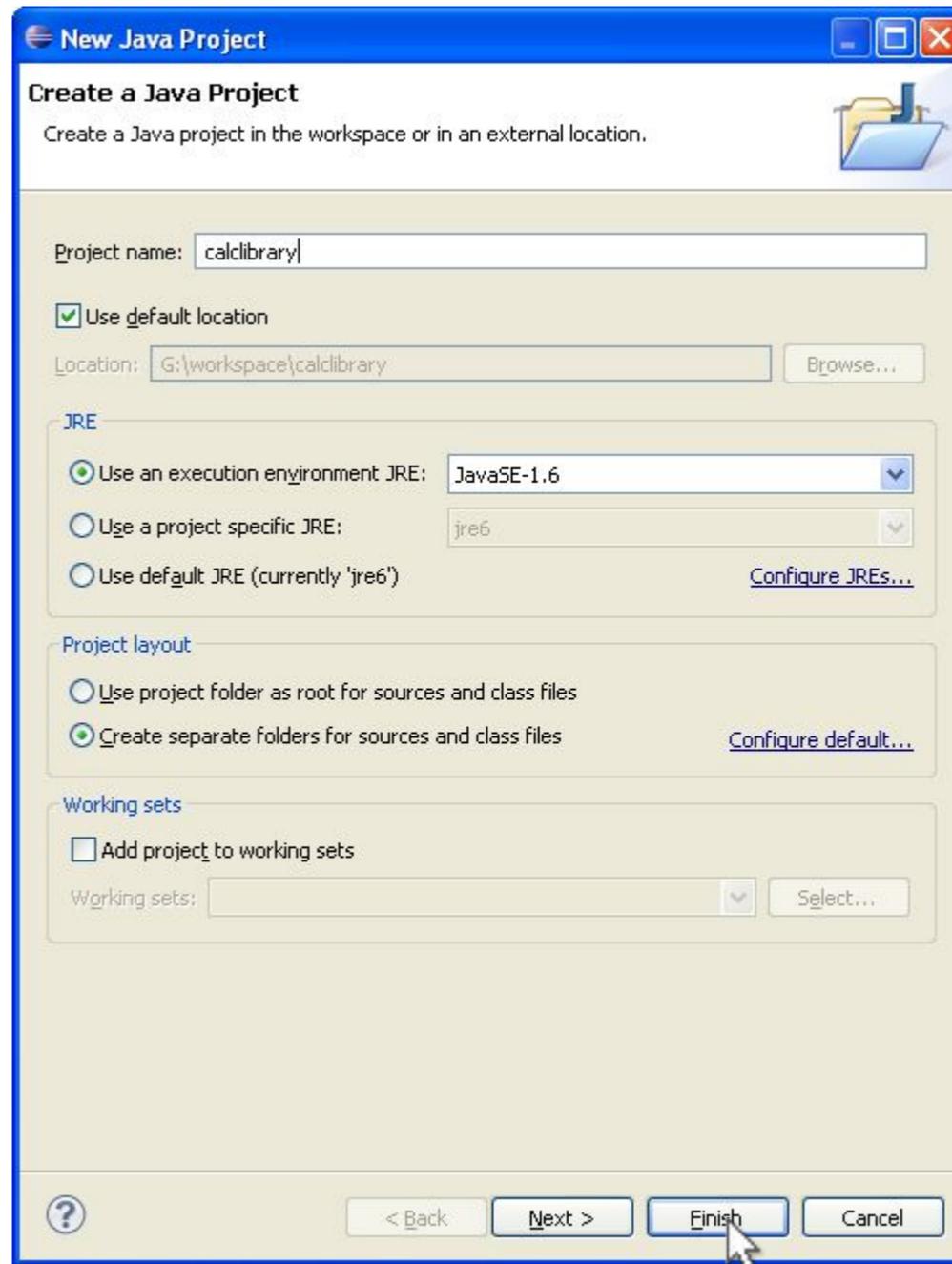
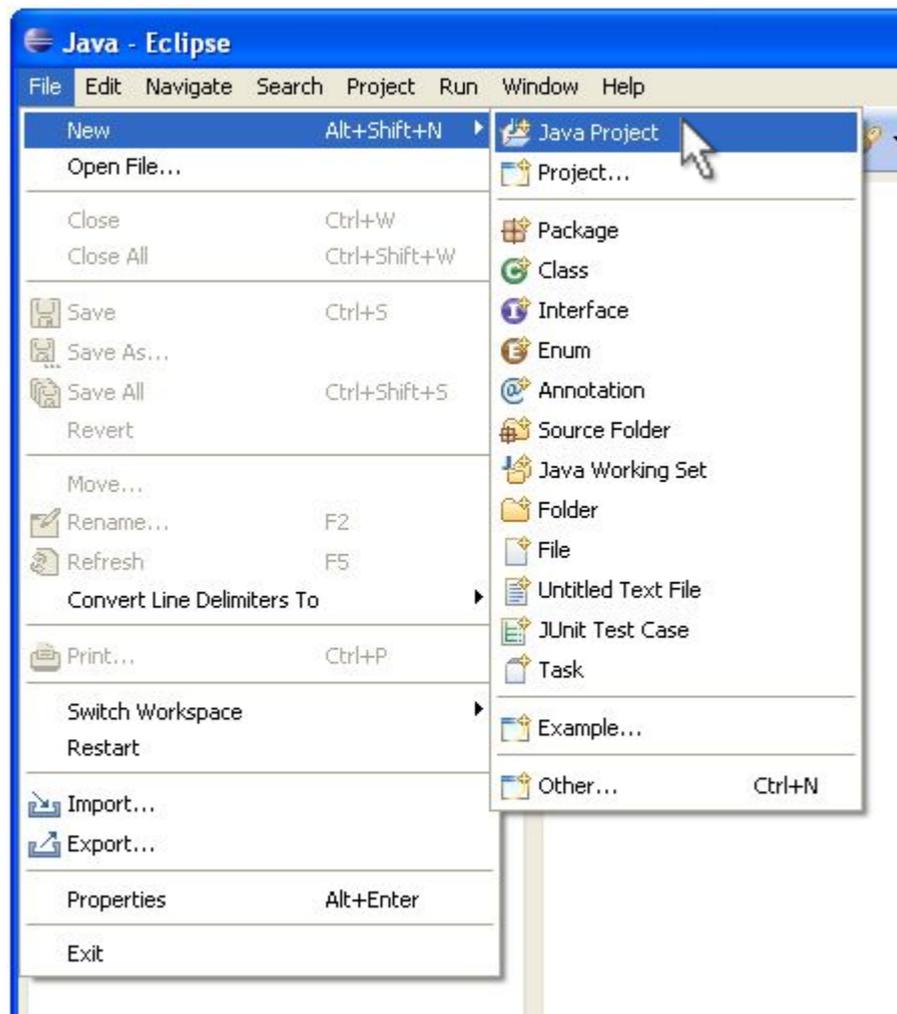
	Вносим изменения в класс HelloCalc и создаём класс Calculator с доступом по умолчанию.
--	--

Запуск приложения

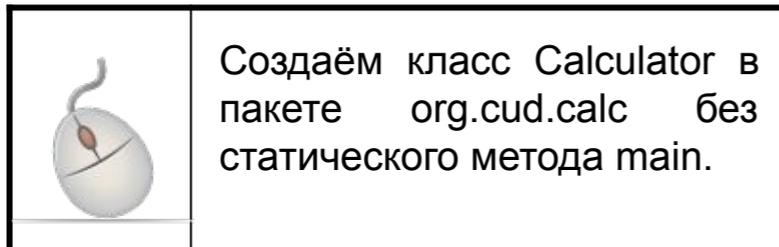
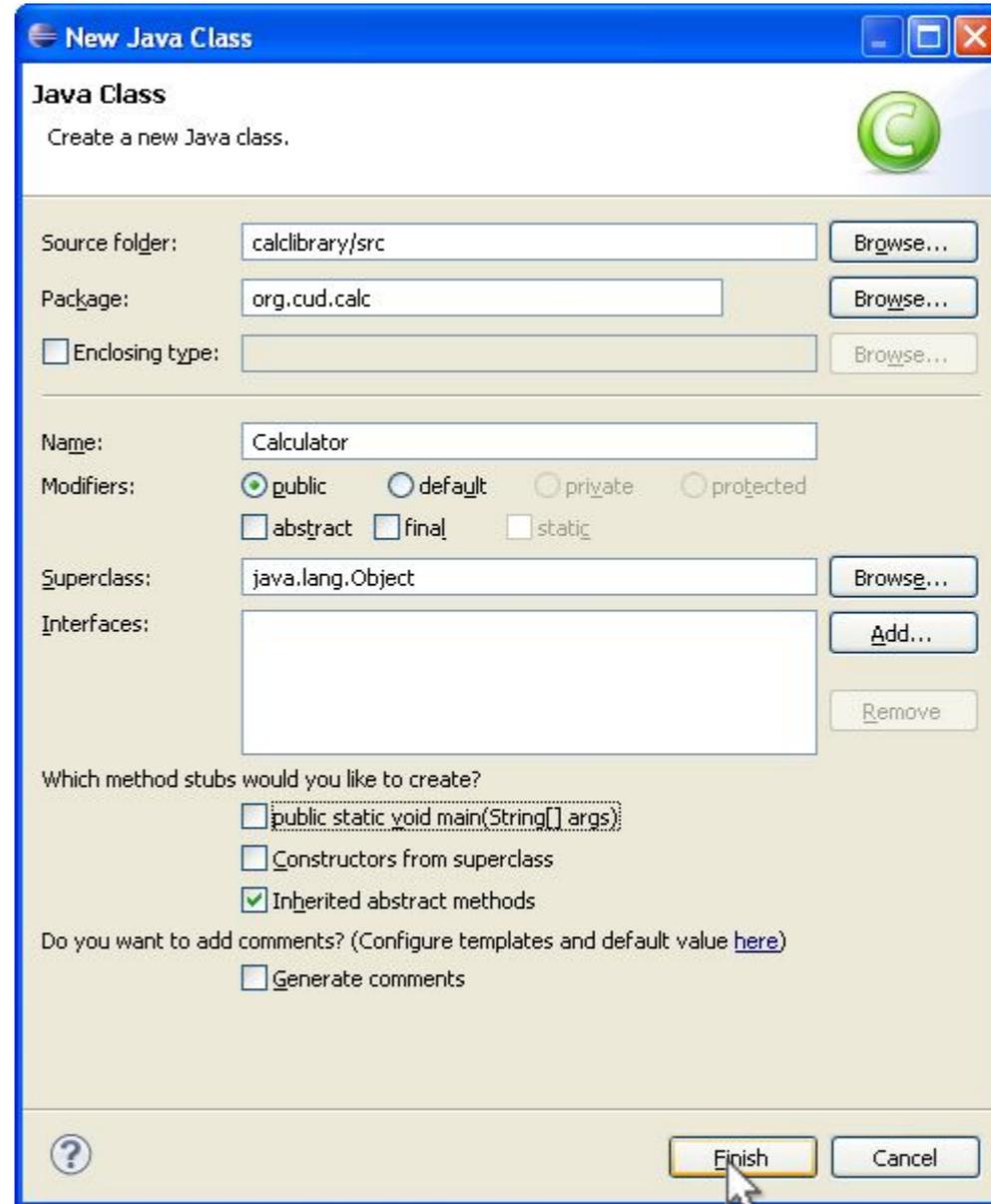
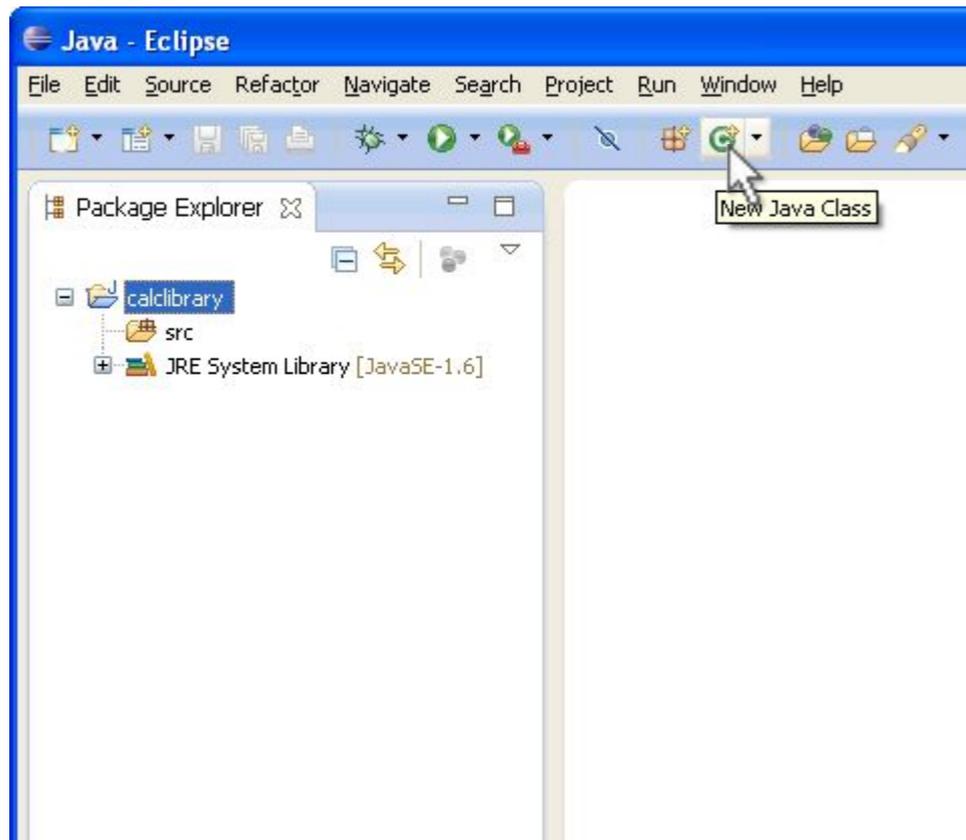


Библиотека

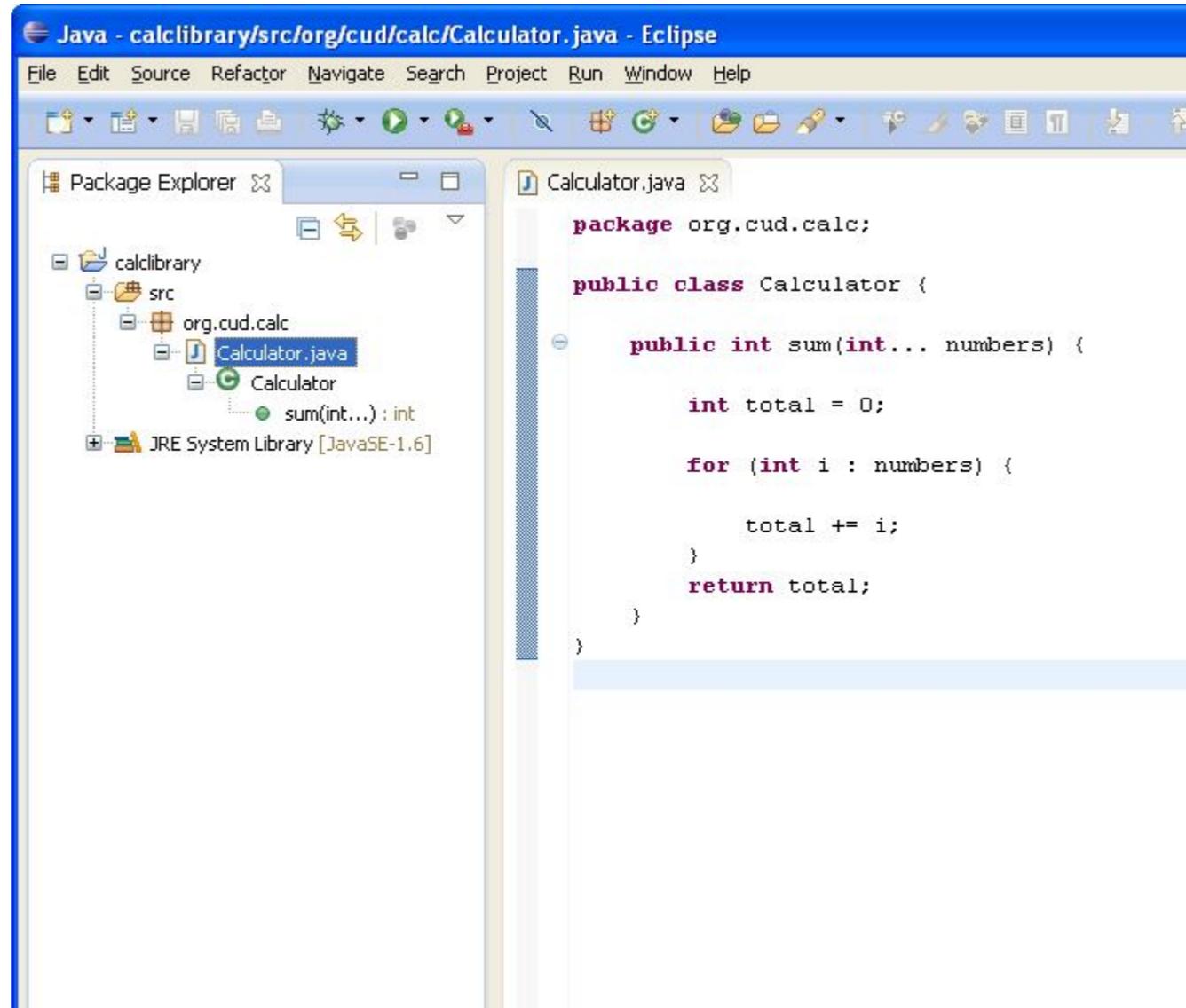
Создание проекта в Eclipse



Создание класса



Внесение изменений



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure: calclibrary > src > org.cud.calc > Calculator.java. The main editor window shows the source code for Calculator.java:

```
package org.cud.calc;

public class Calculator {

    public int sum(int... numbers) {

        int total = 0;

        for (int i : numbers) {

            total += i;

        }

        return total;

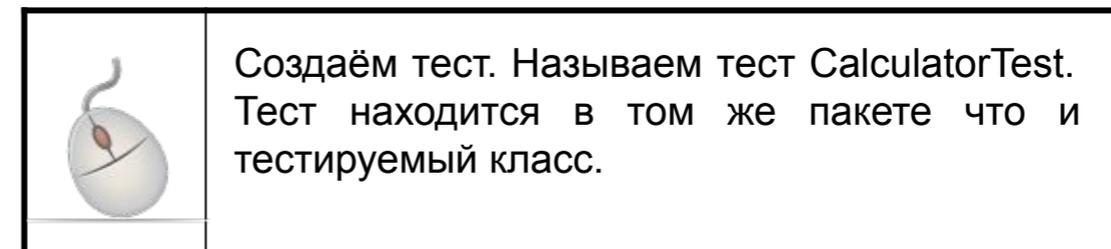
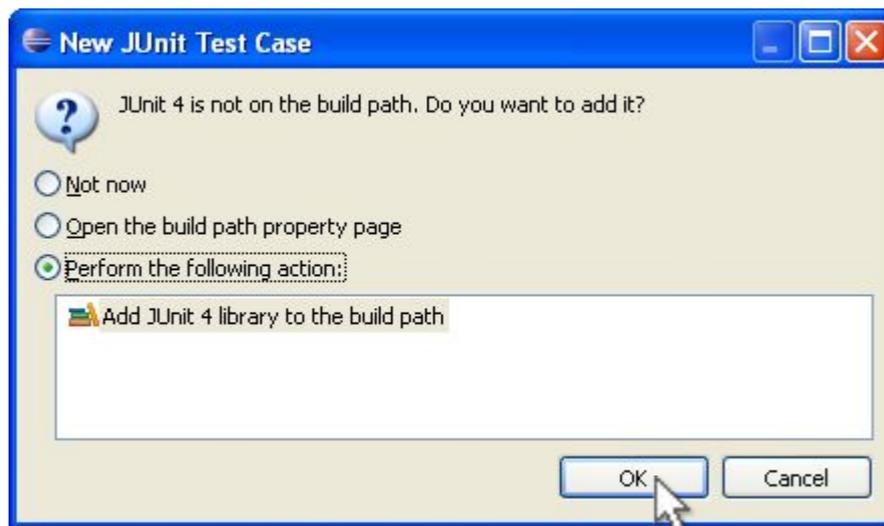
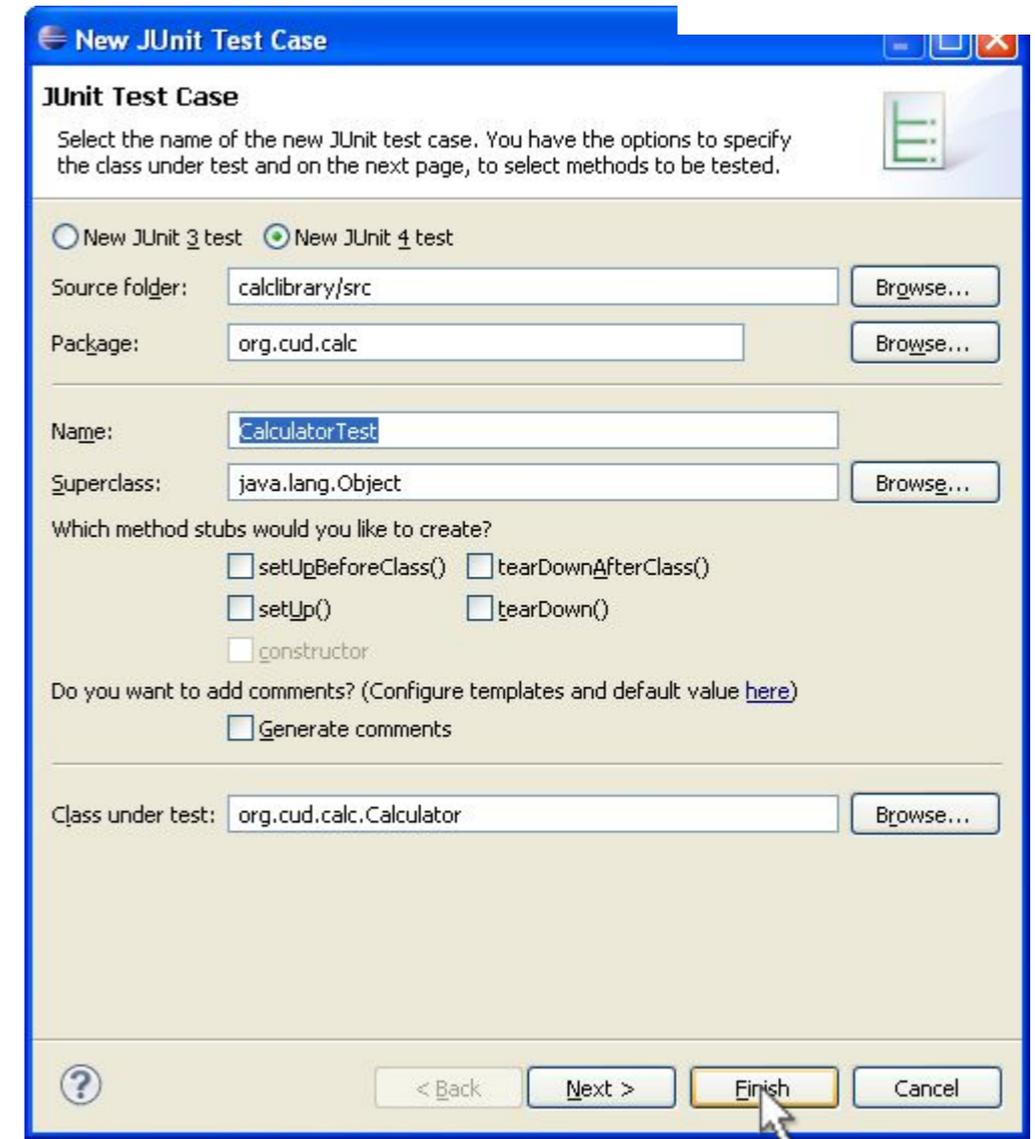
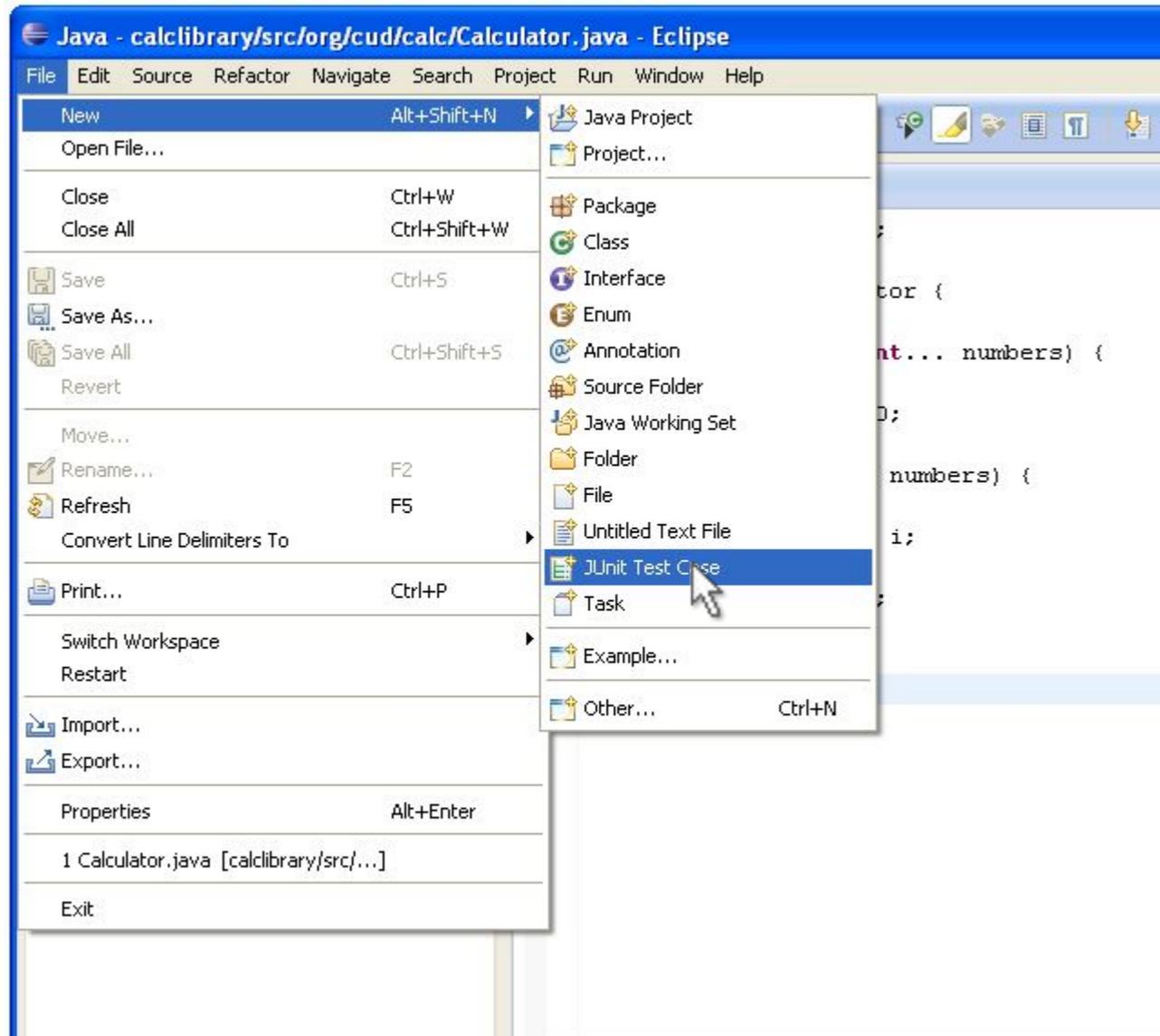
    }

}
```

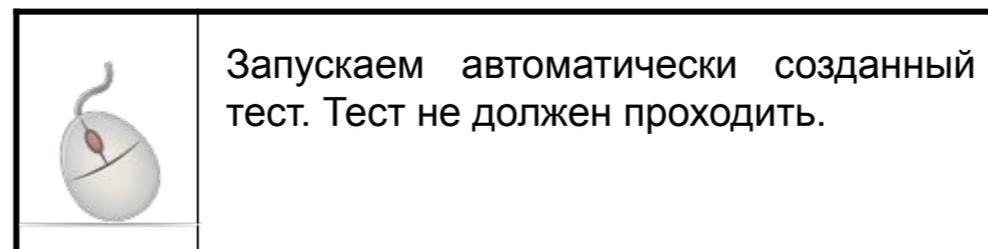
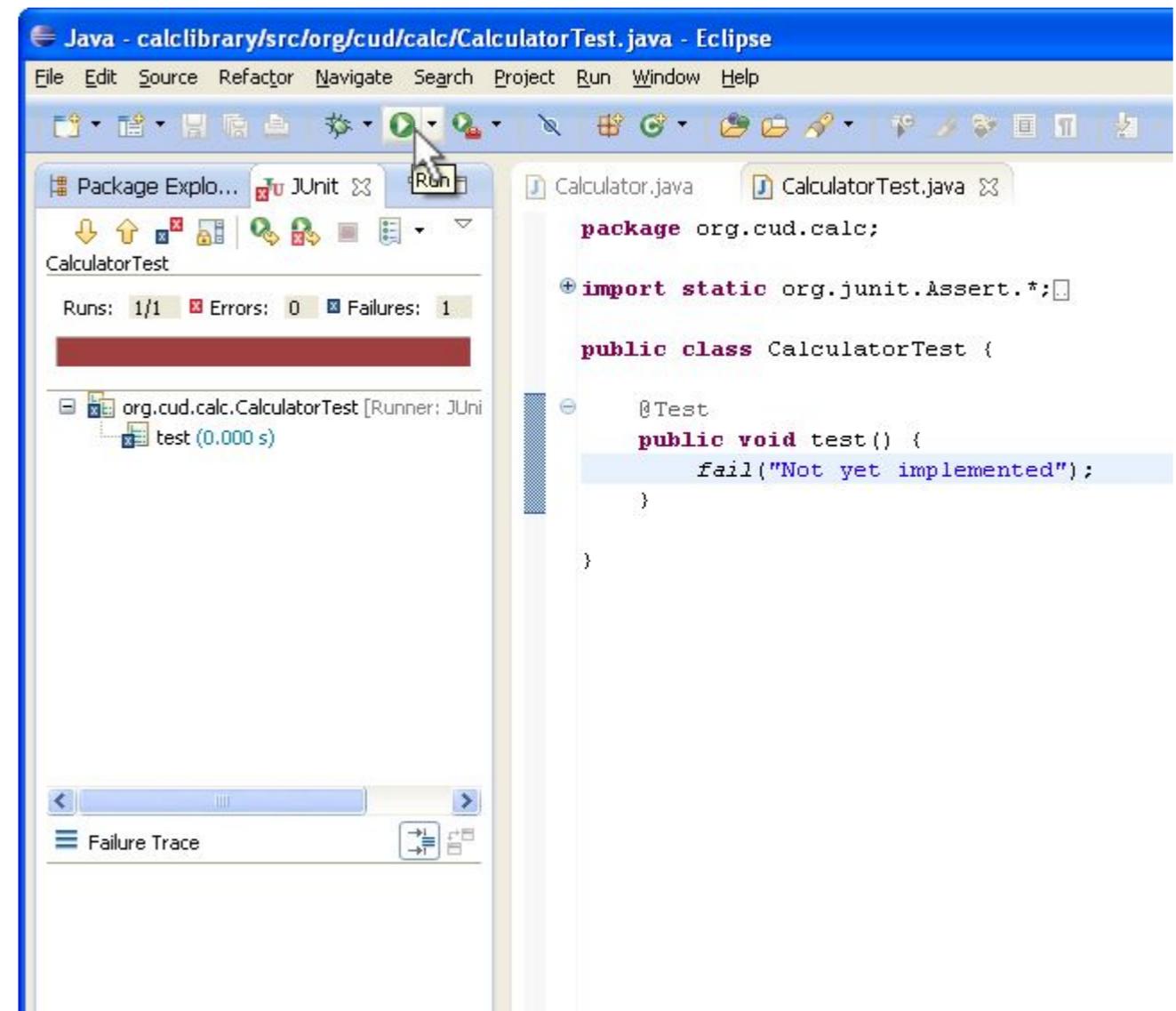
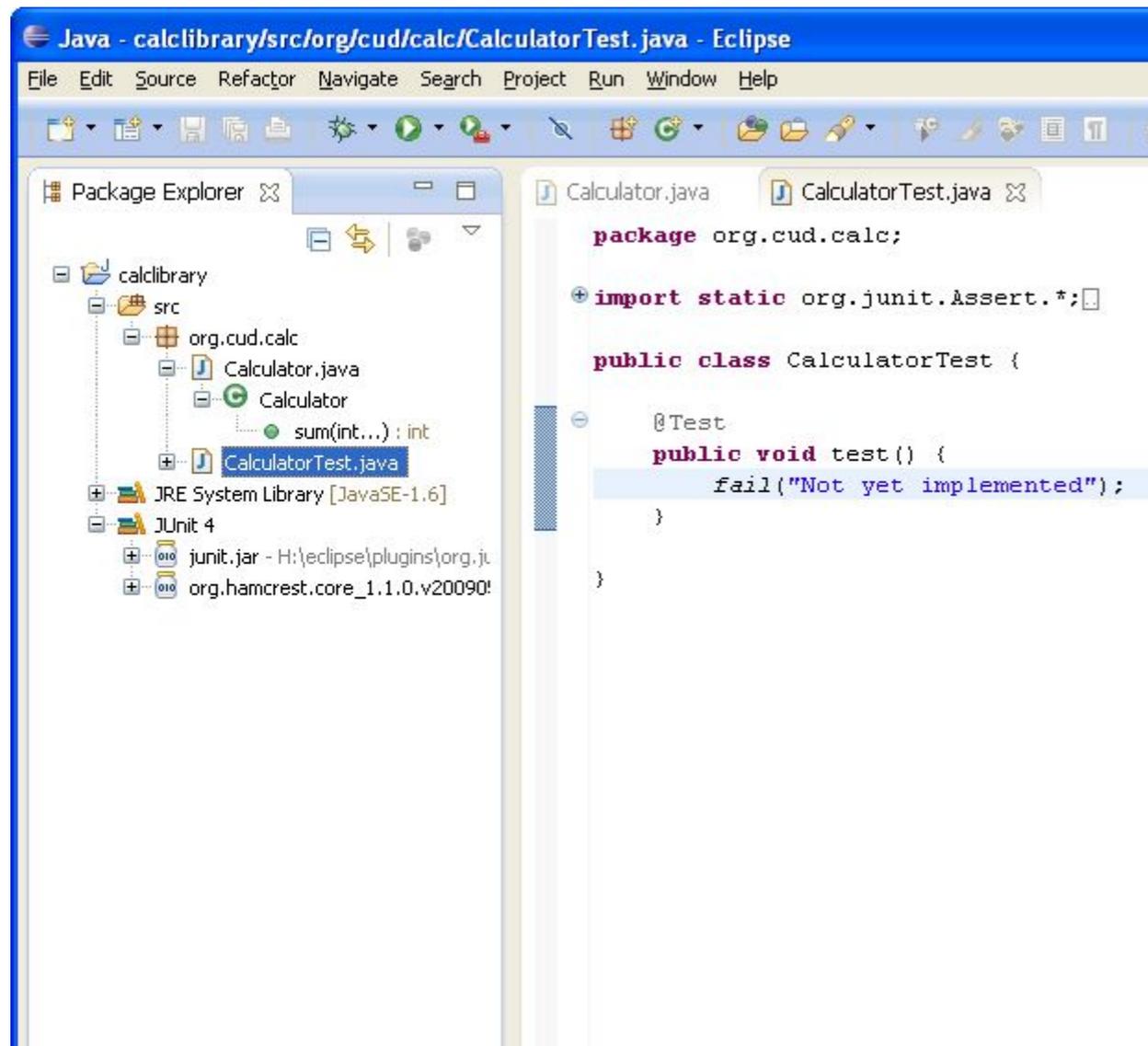
	Вносим изменения в класс Calculator. Добавляем метод для подсчёта суммы чисел.
---	--

Тестирование библиотеки с JUnit

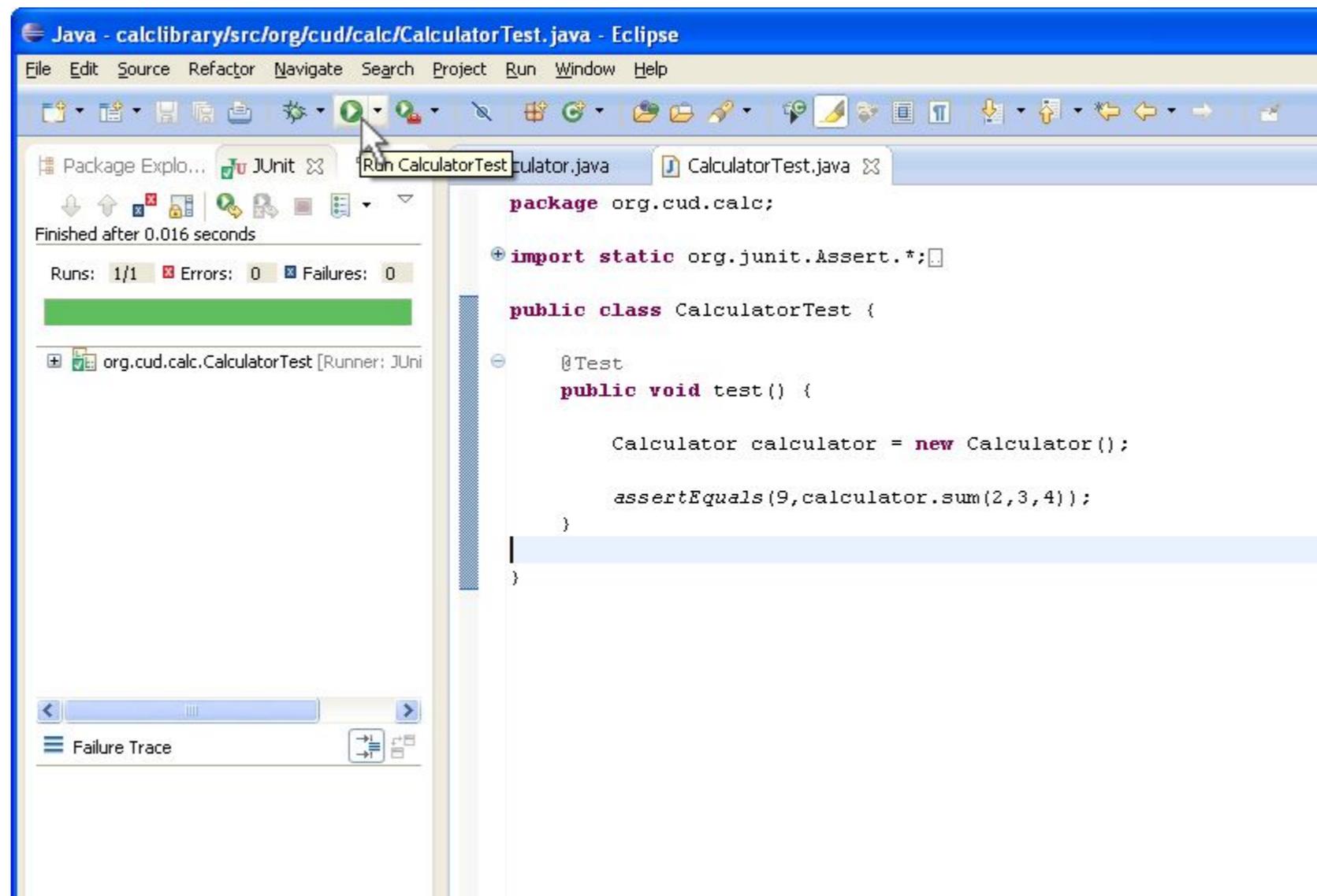
Создаём тест



Автоматически созданный тест

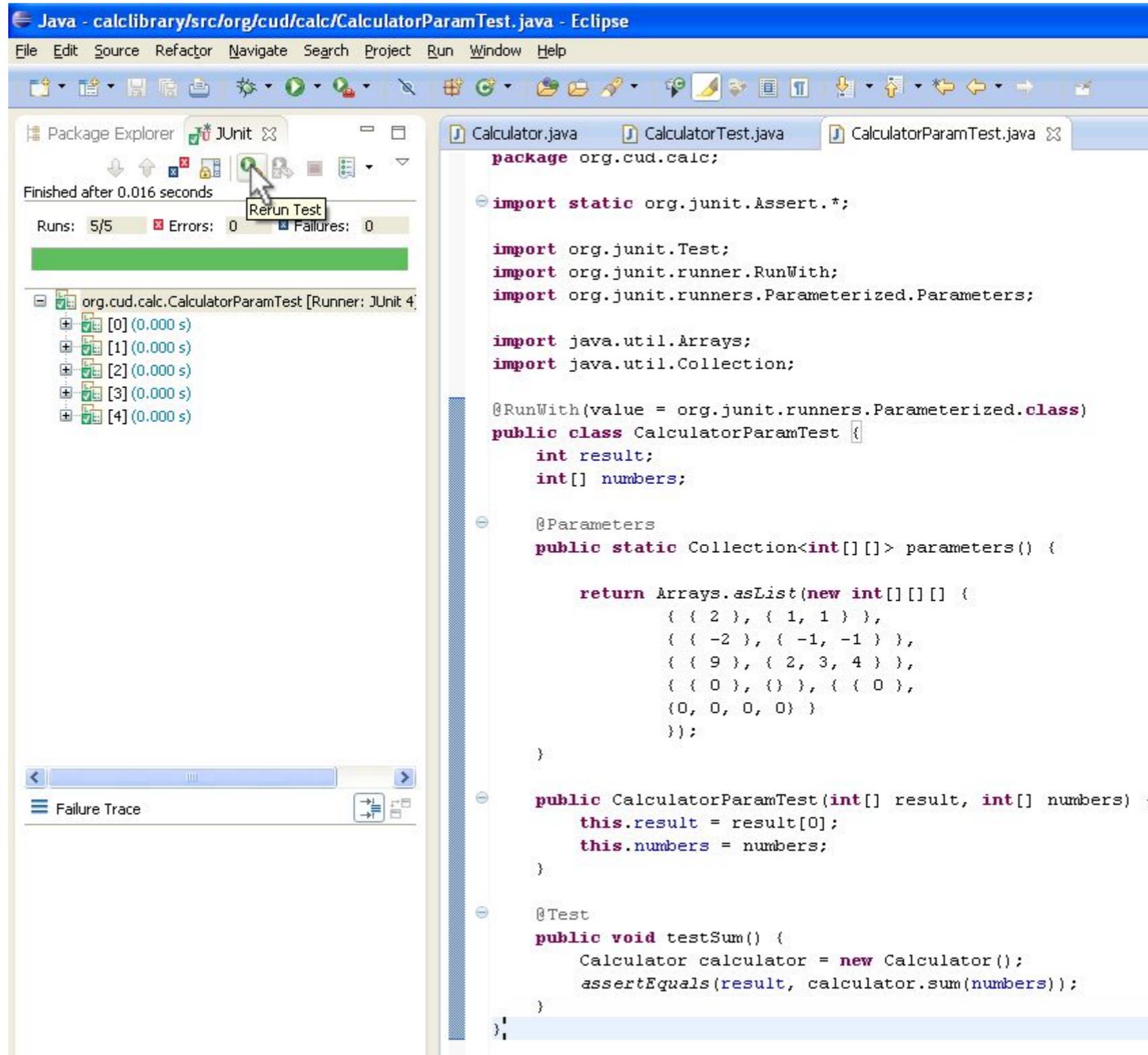


Тест



 Меняем тест. Метод для подсчёта суммы чисел проходит этот тест.

Параметризованный тест



Java - calclibrary/src/org/cud/calc/CalculatorParamTest.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer JUnit

Finished after 0.016 seconds

Runs: 5/5 Errors: 0 Failures: 0

org.cud.calc.CalculatorParamTest [Runner: JUnit 4]

- [0] (0.000 s)
- [1] (0.000 s)
- [2] (0.000 s)
- [3] (0.000 s)
- [4] (0.000 s)

```
package org.cud.calc;

import static org.junit.Assert.*;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.junit.runners.Parameterized.Parameters;

import java.util.Arrays;
import java.util.Collection;

@RunWith(value = org.junit.runners.Parameterized.class)
public class CalculatorParamTest {
    int result;
    int[] numbers;

    @Parameters
    public static Collection<int[] []> parameters() {

        return Arrays.asList(new int[] [] [] {
            { { 2 }, { 1, 1 } },
            { { -2 }, { -1, -1 } },
            { { 9 }, { 2, 3, 4 } },
            { { 0 }, { } }, { { 0 },
            { 0, 0, 0, 0 } }
        });
    }

    public CalculatorParamTest(int[] result, int[] numbers) {
        this.result = result[0];
        this.numbers = numbers;
    }

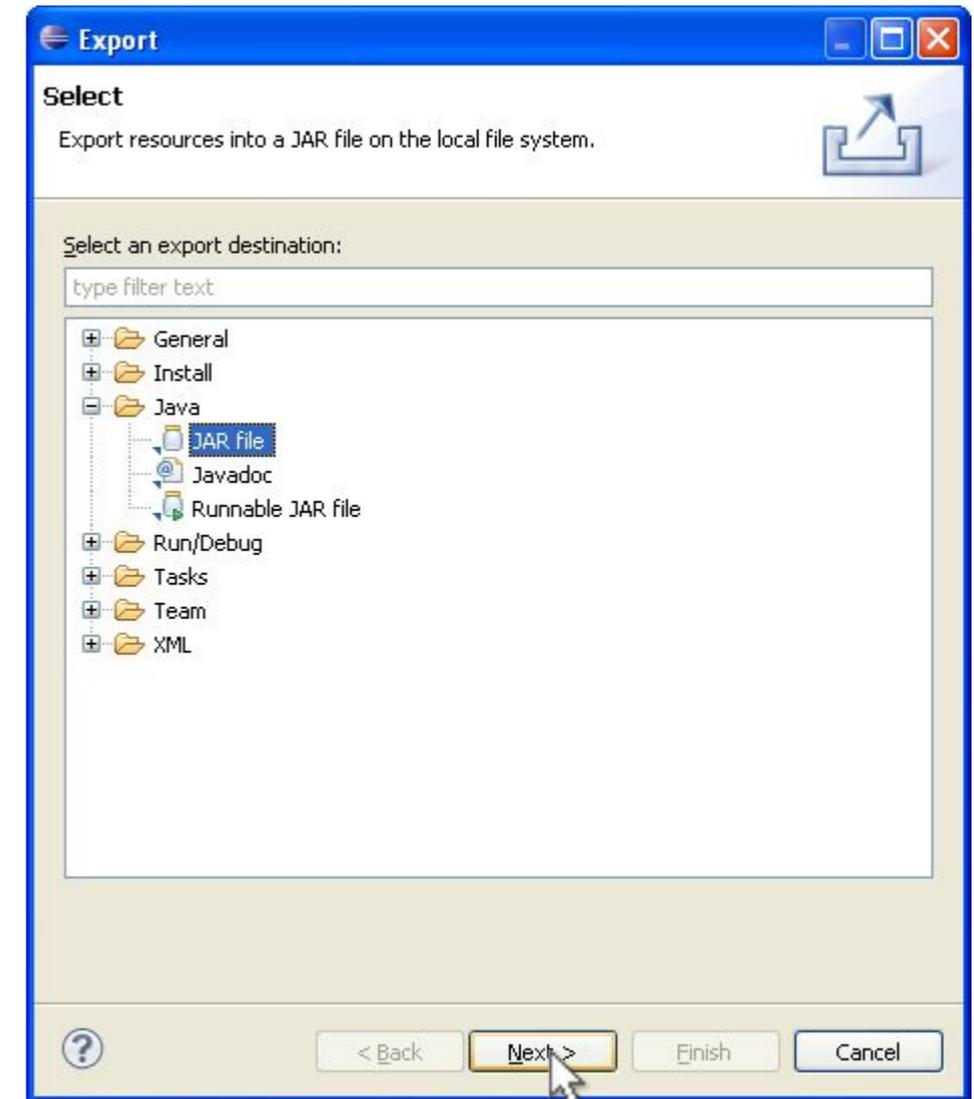
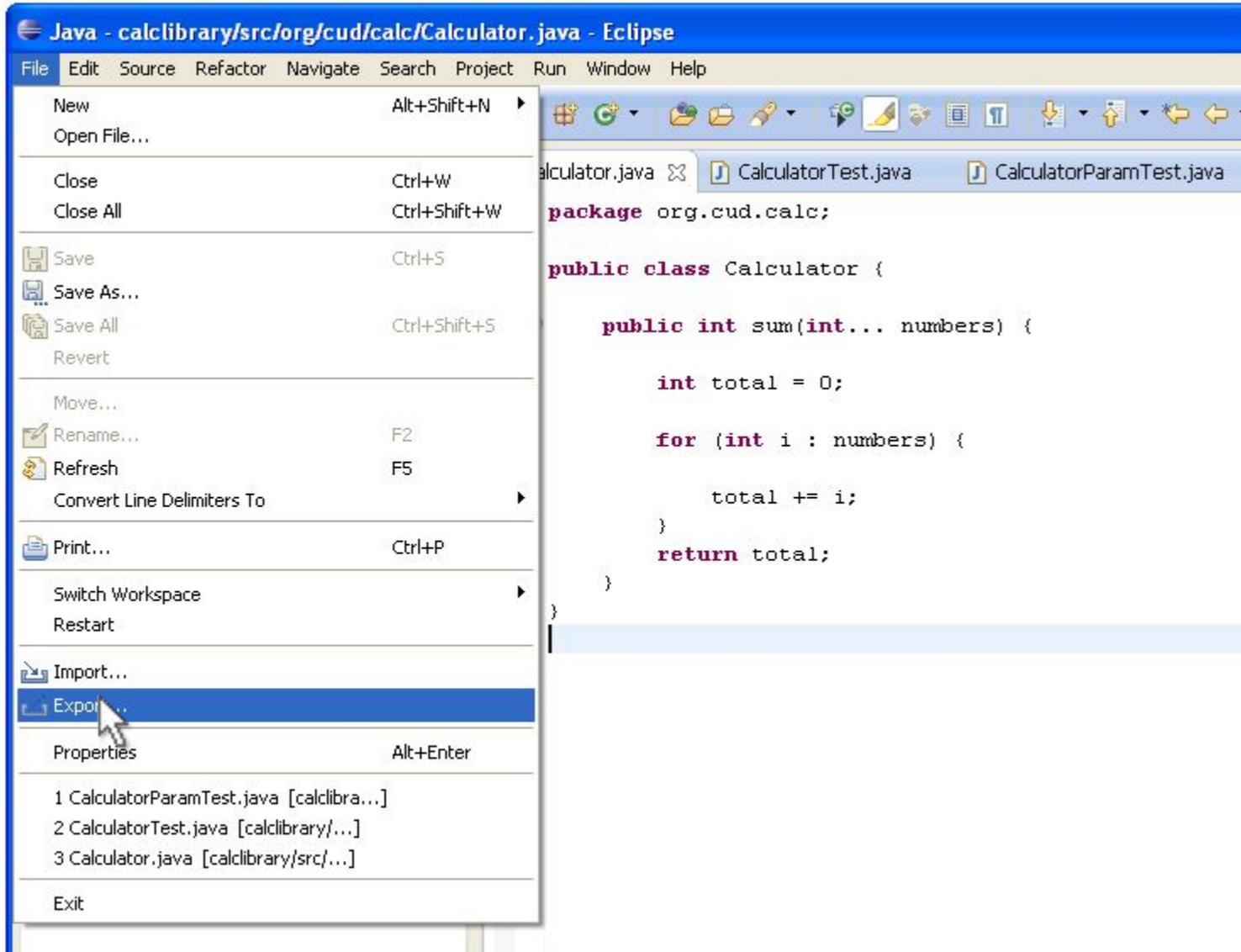
    @Test
    public void testSum() {
        Calculator calculator = new Calculator();
        assertEquals(result, calculator.sum(numbers));
    }
}
```



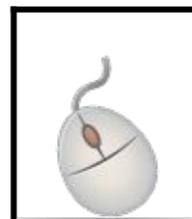
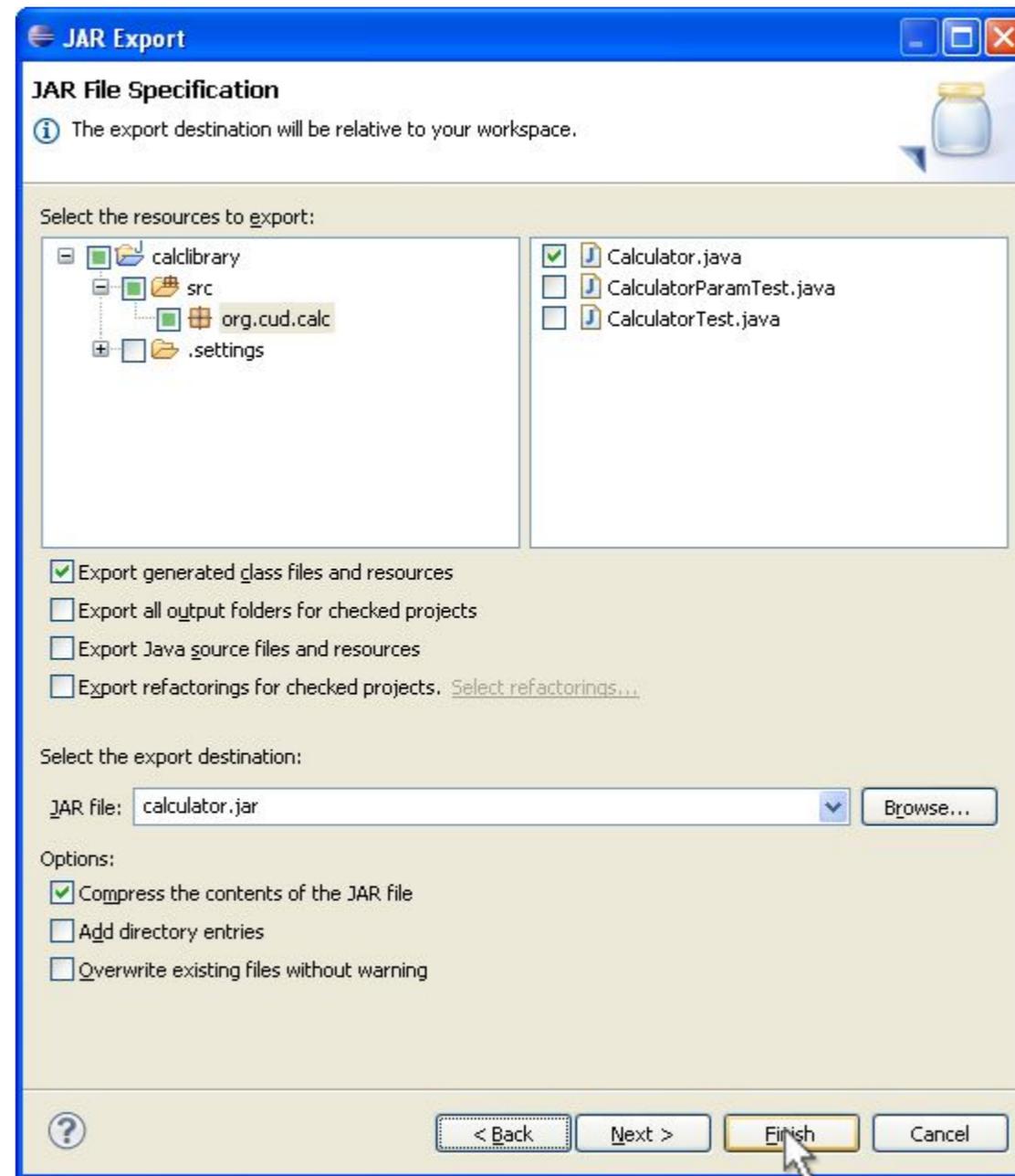
Создаём параметризованный тест. Такой тест позволяет проверить сразу много вариантов.

Создание JAR архива

Создание JAR архива

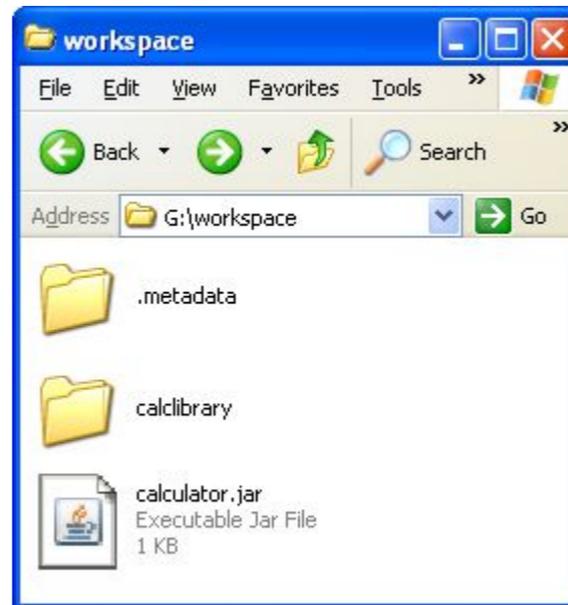


Выбор содержимого



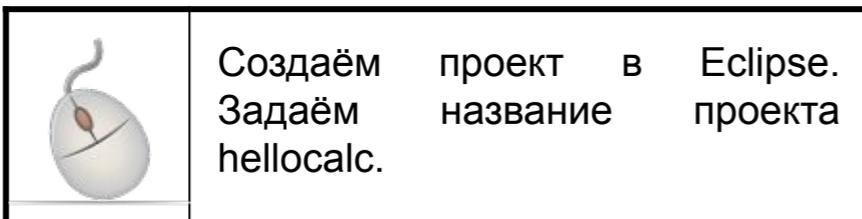
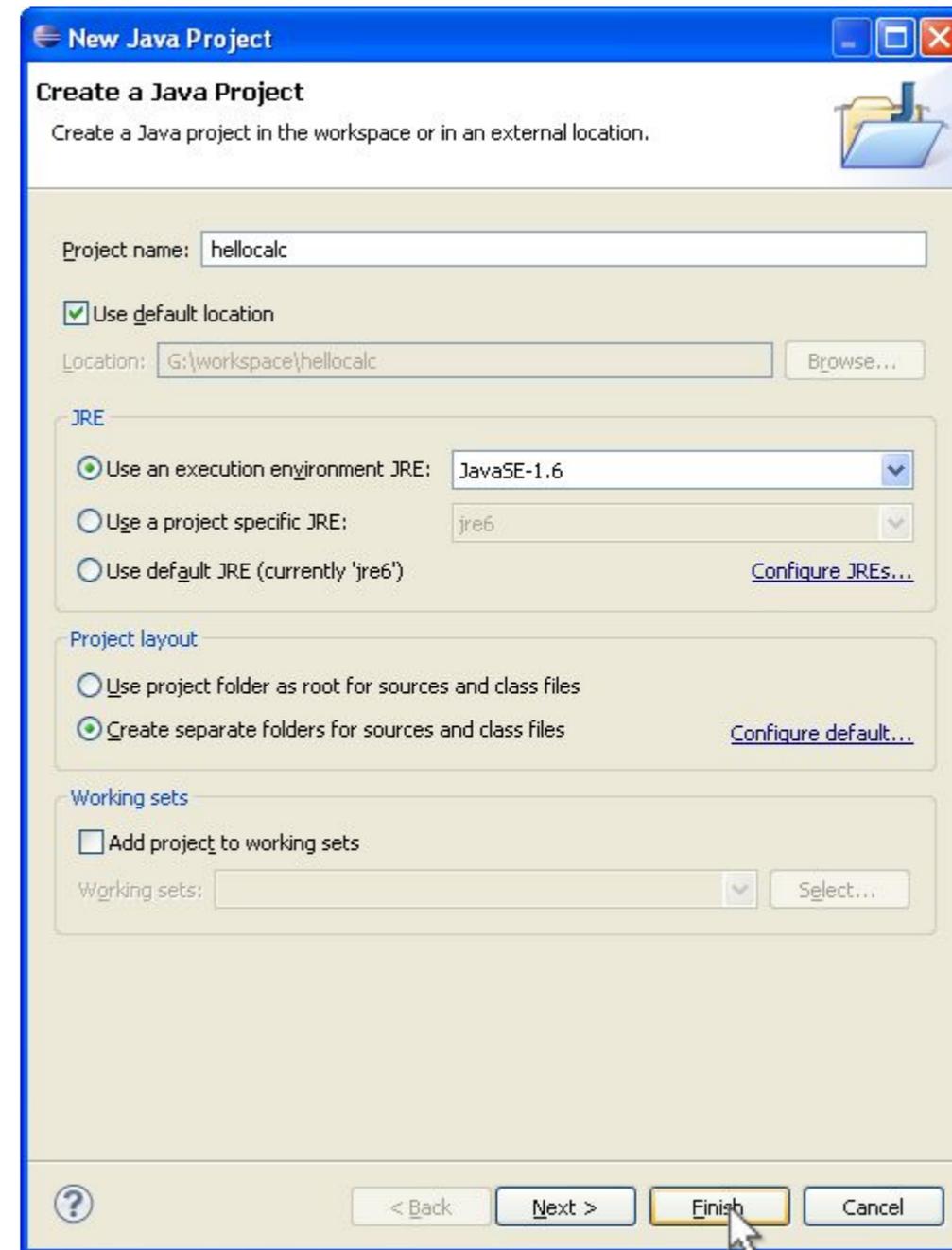
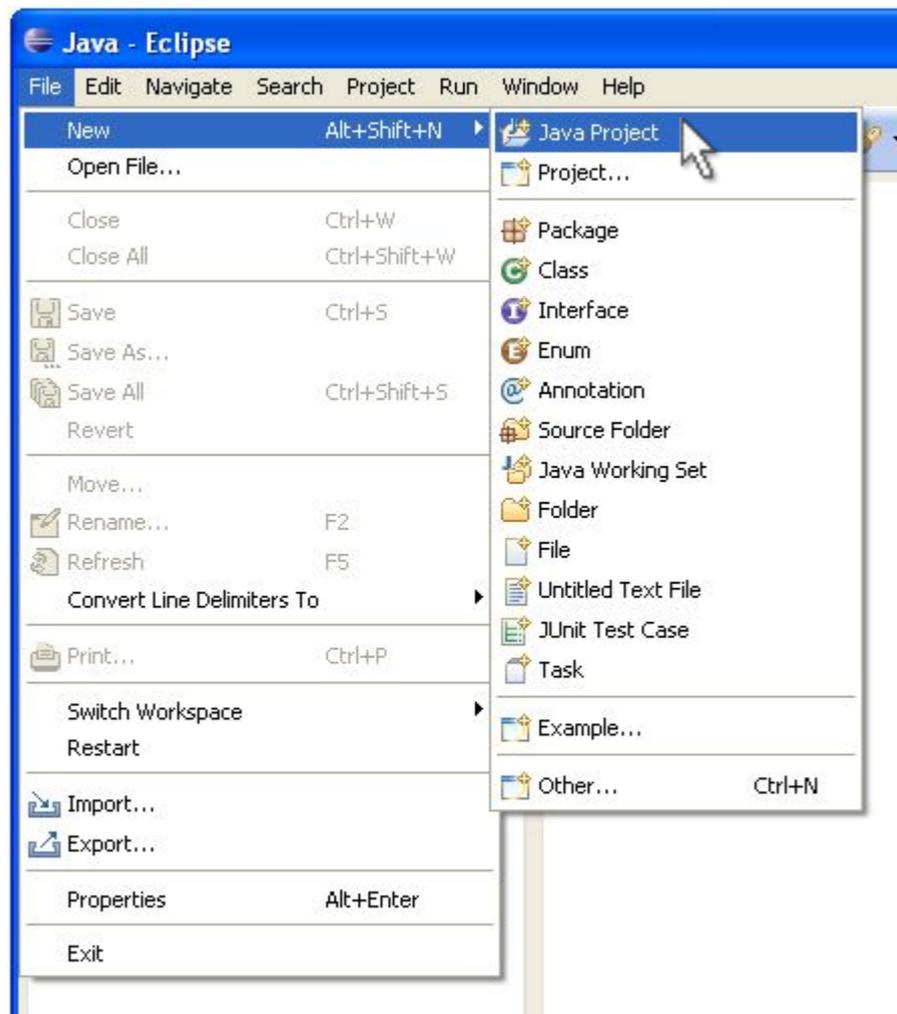
В JAR архив включим только class файл для Calculator. Тесты включать не будем.

Созданный JAR архив

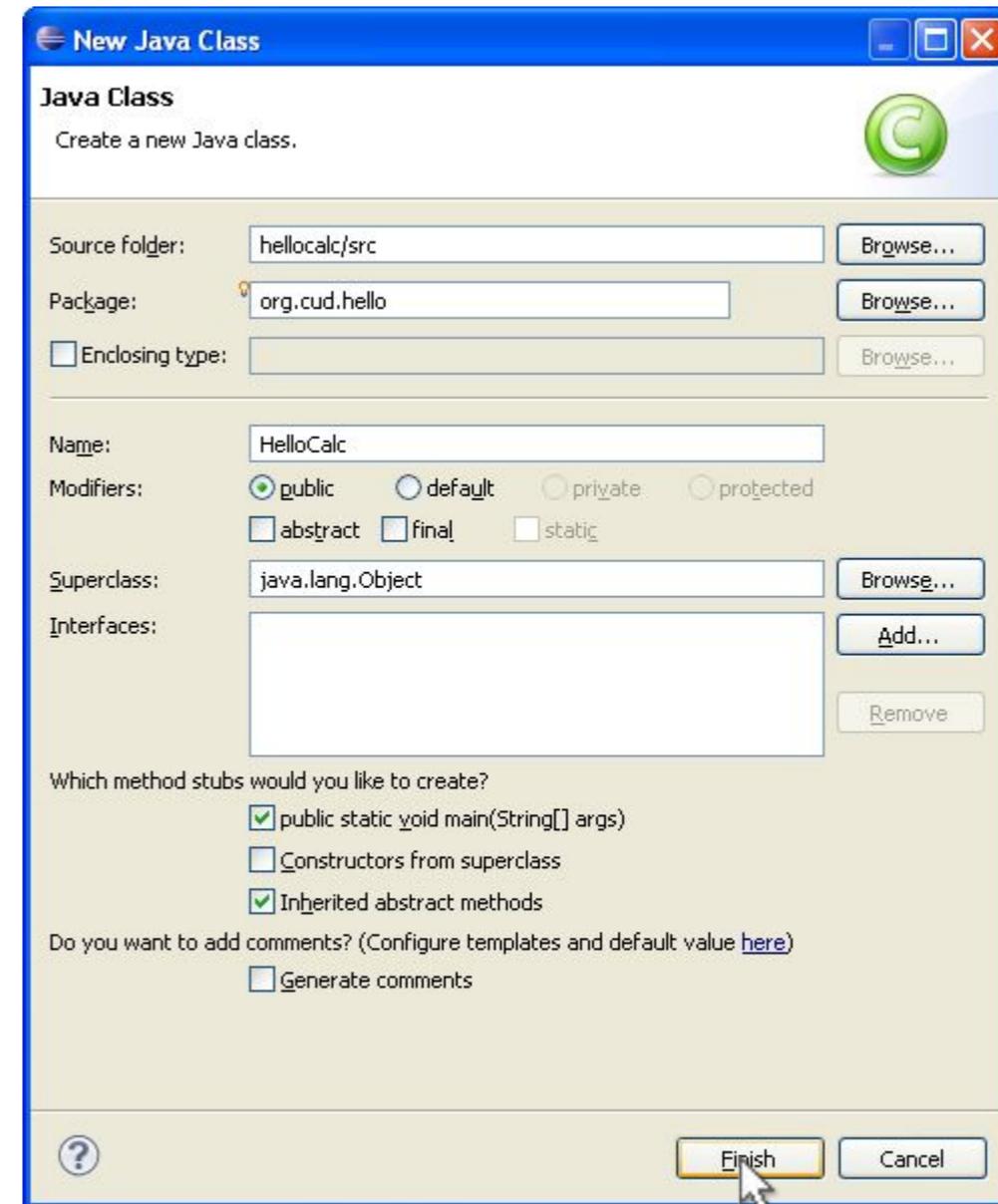
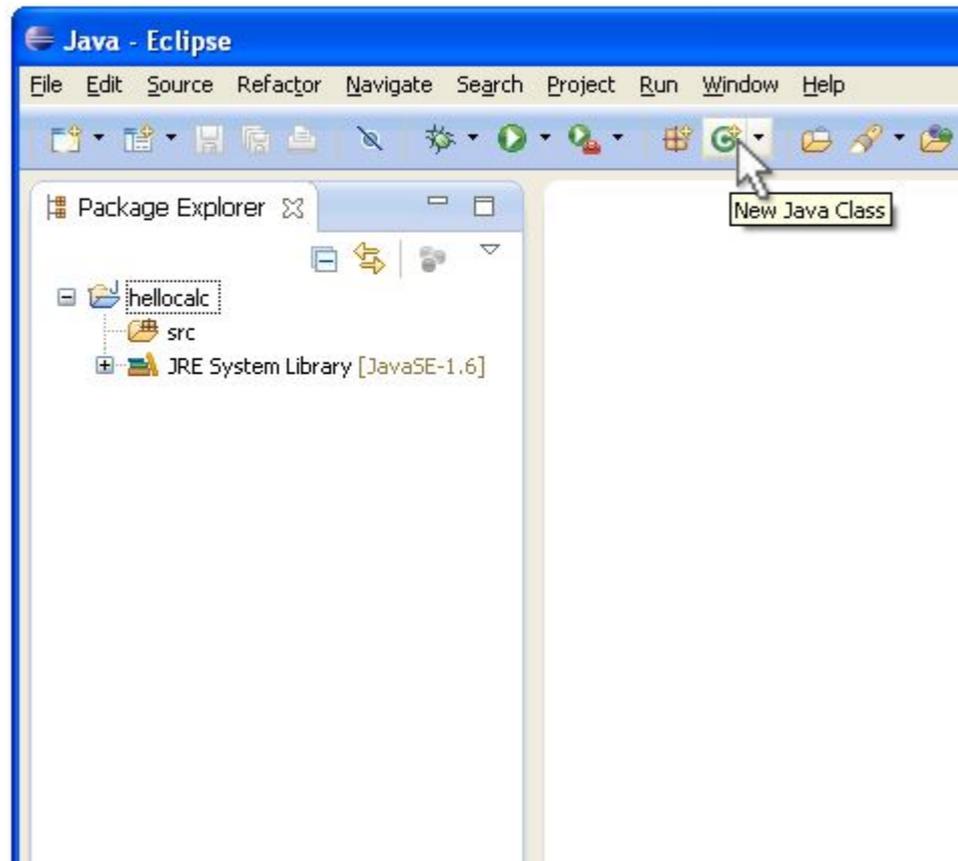


Использование JAR архива

Создание проекта в Eclipse

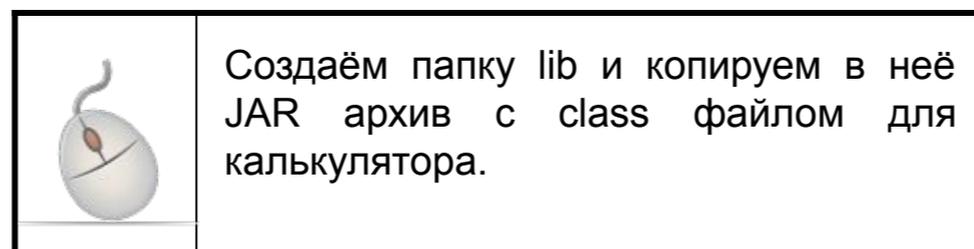
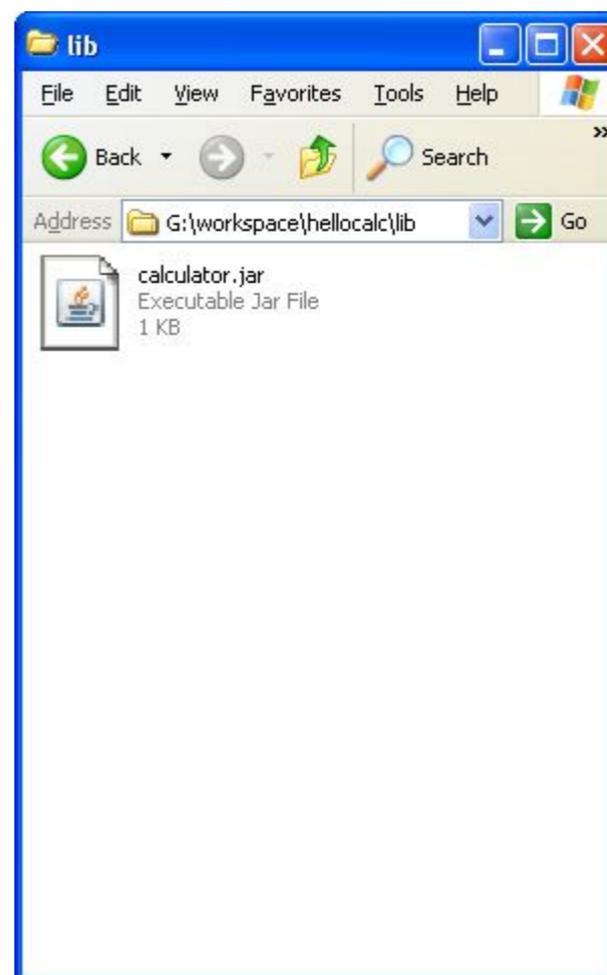


Создание класса

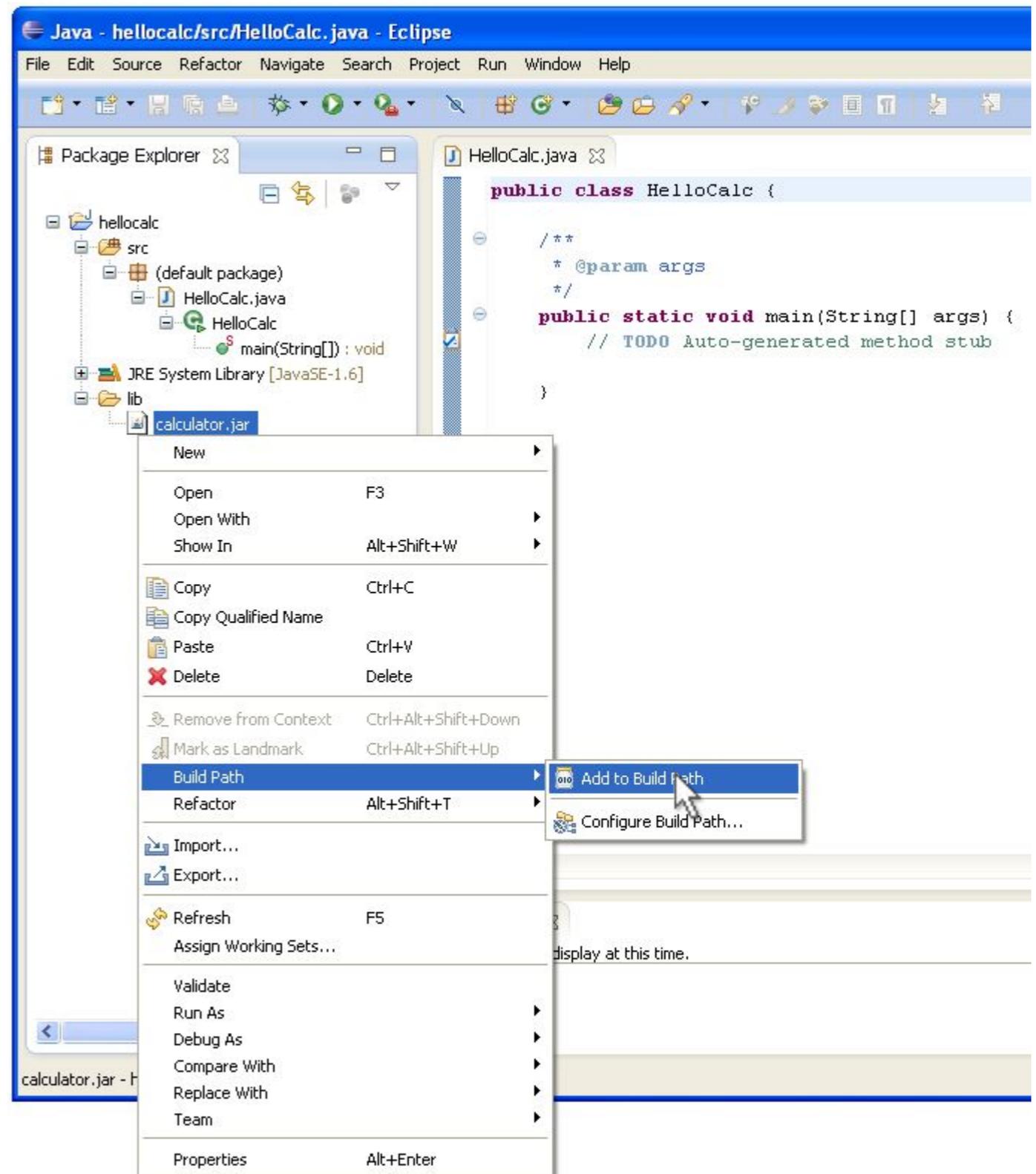
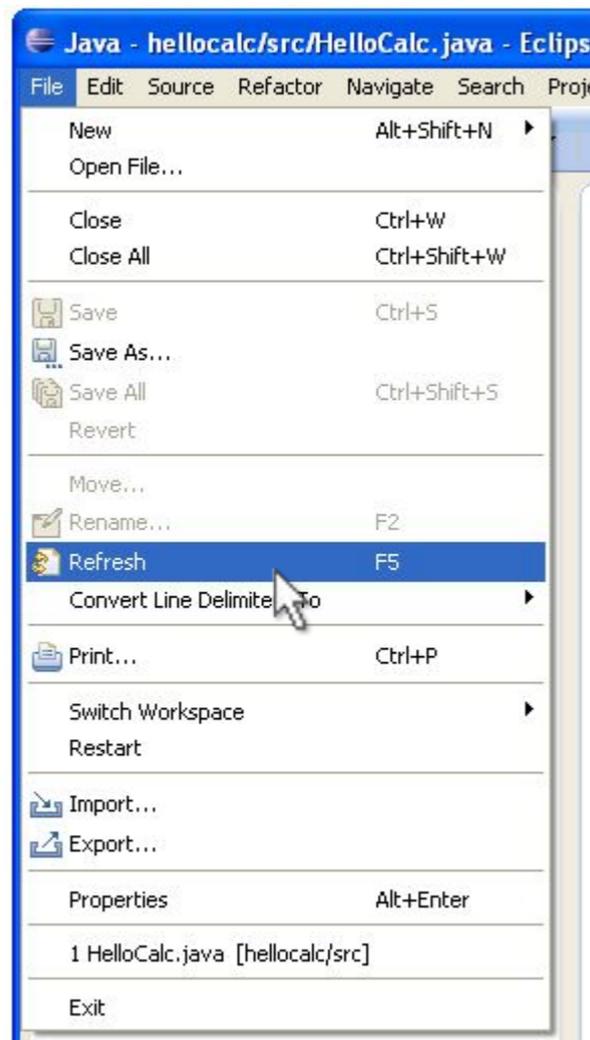


 Создаём класс HelloCalc в пакете org.cud.hello со статическим методом main.

Копирование JAR архива

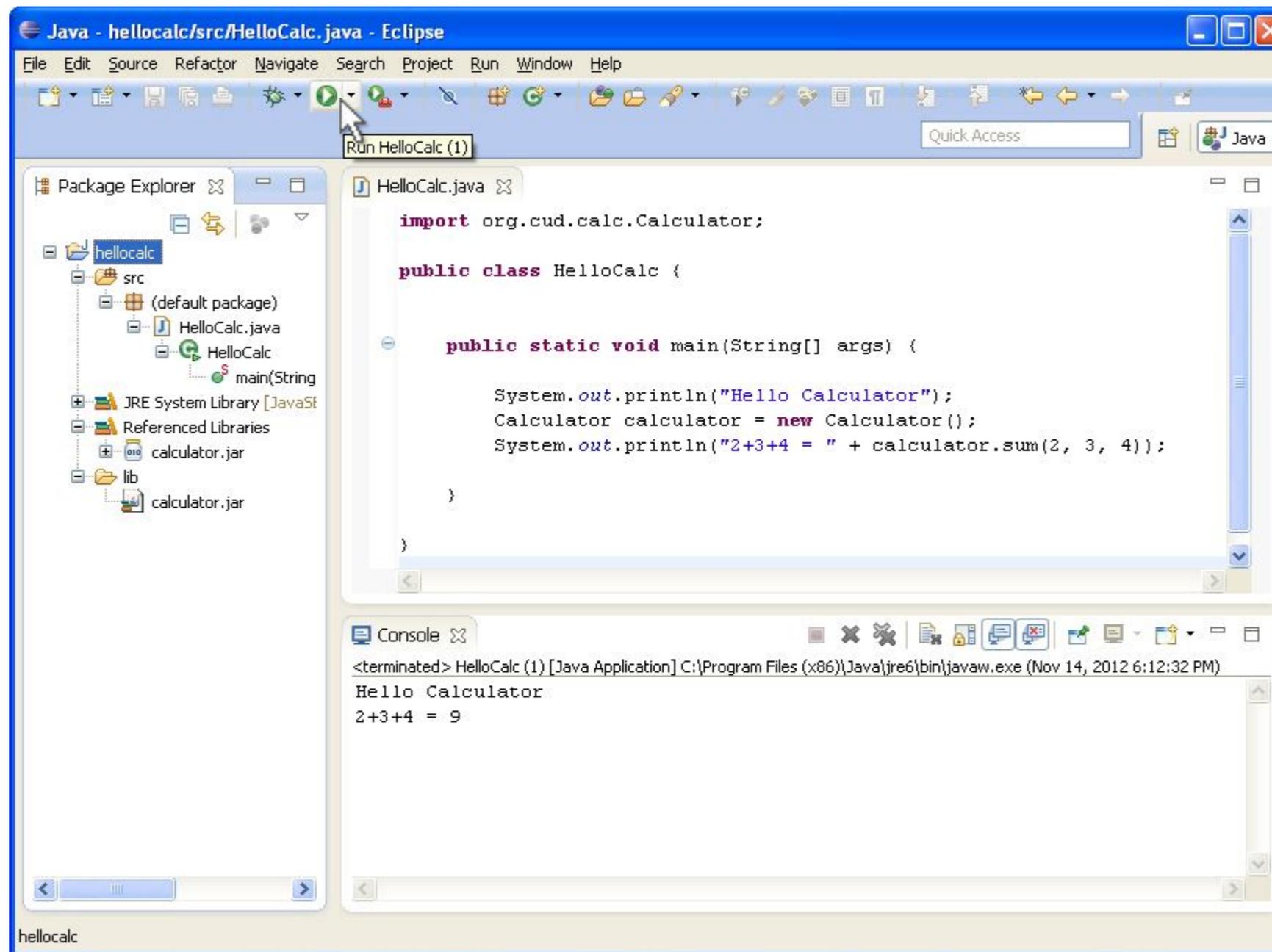


Добавление JAR архива в BuildPath



 Выбираем проект в Package Explorer и нажимаем F5. Добавляем в buildpath jar файл с классом Calculator.

Вносим изменения и запускаем



 Добавляем объявление импорта для org.cud.calc.Calculator. Вносим изменения в класс HelloCalc и запускаем.