СОВРЕМЕННЫЕ ТЕХНОЛОГИИ РАЗРАБОТКИ ПО

Практикум:

Модульное тестирование с JUnit

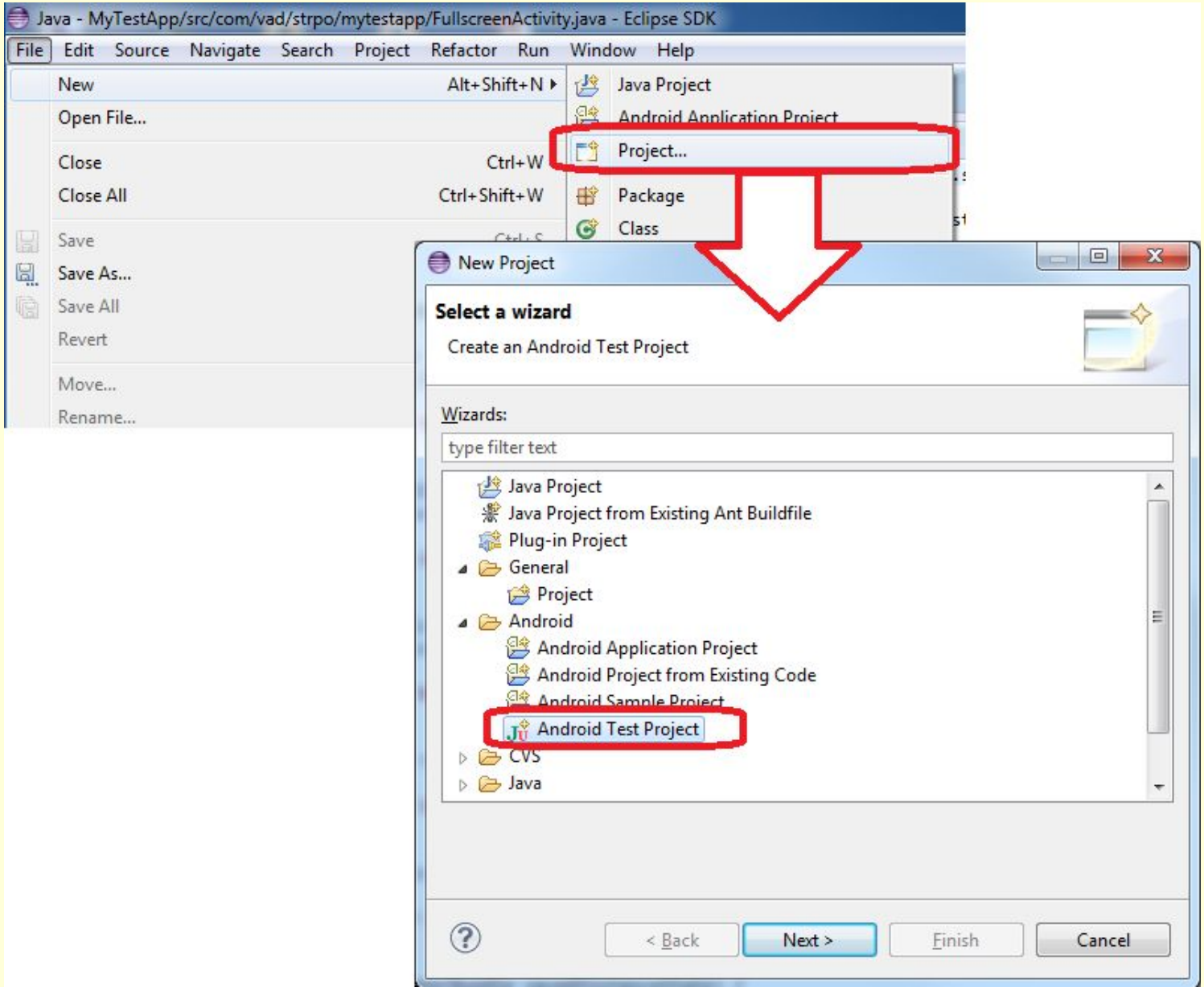
JUnit

- Библиотека для модульного тестирования
  - поставляется в виде jar-файла
  - в составе большинства IDE для Java
- JUnit запускает пакеты тестов и выдаёт отчёт о результатах
- Для каждого теста выполняет действия:
  - setUp() – настройка окружения
  - запуск тестового метода
    - тестовый метод проверяет какой-то аспект поведения: вызывает тестируемый метод класса и проверяет утверждения относительно результатов
  - tearDown() – уничтожение окружения

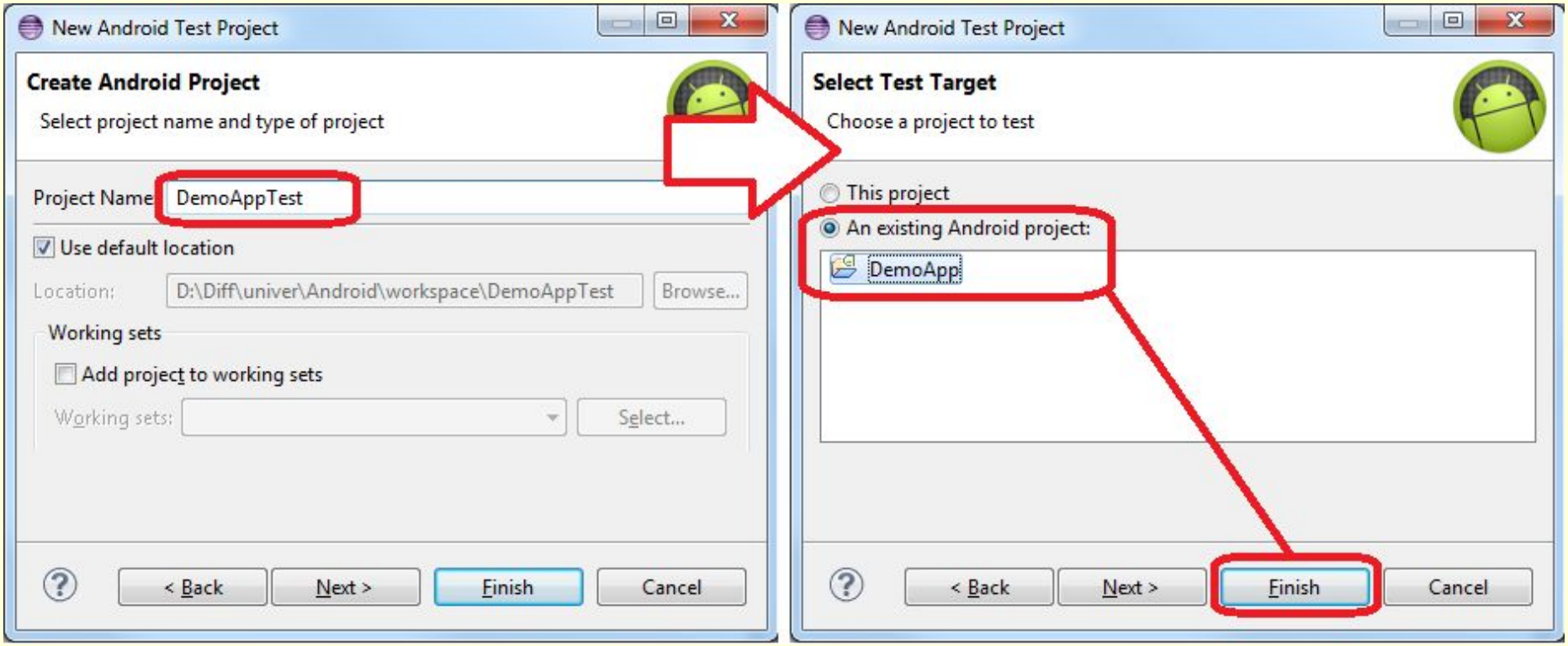
Проект с тестами

- Создаём новый проект
  - “Java Project” / “Android Test Project”
- Определяем зависимость от тестируемого проекта
- Создаём модули с тестами
  - Best practices: параллельная структура пакетов (дублирует структуру тестируемого проекта)

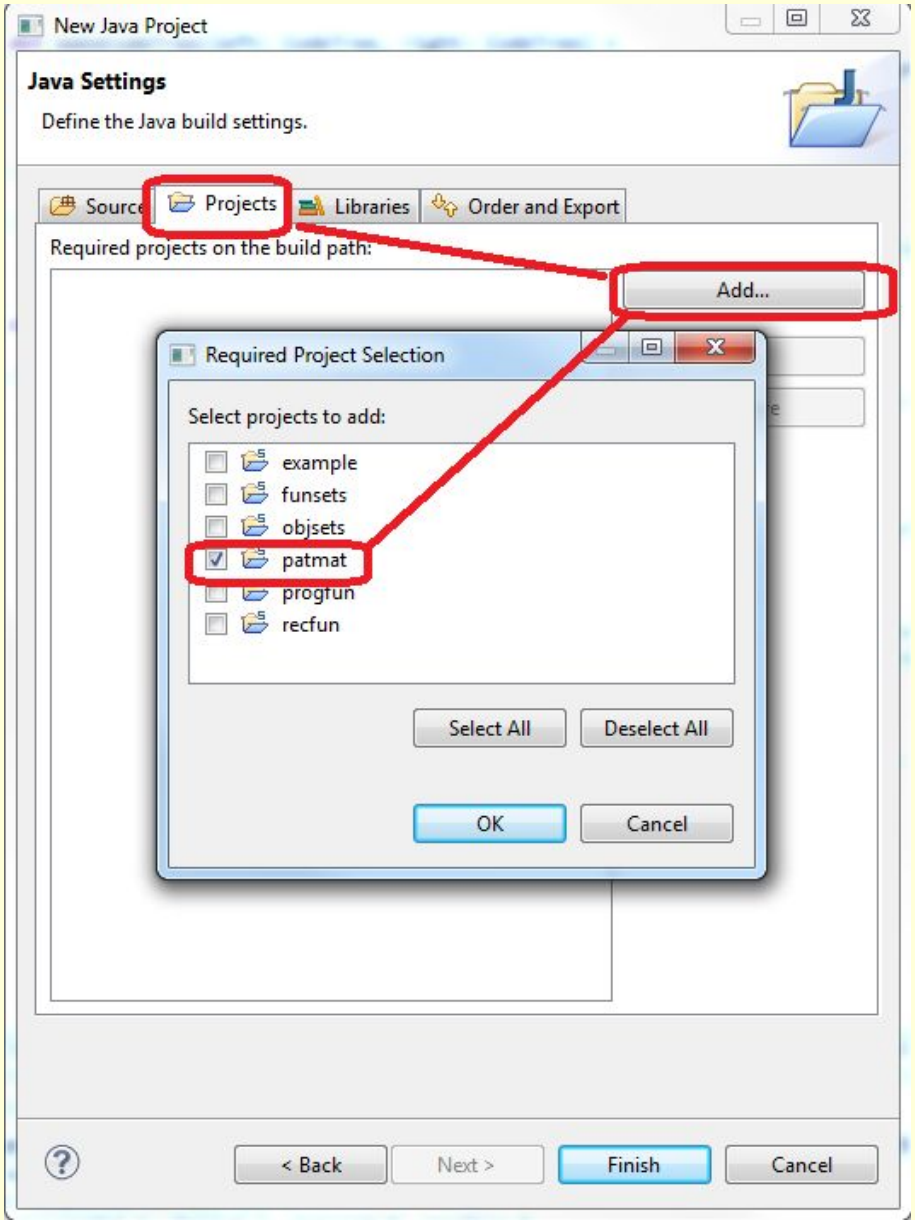
Проект с тестами: Android



Проект с тестами: Android



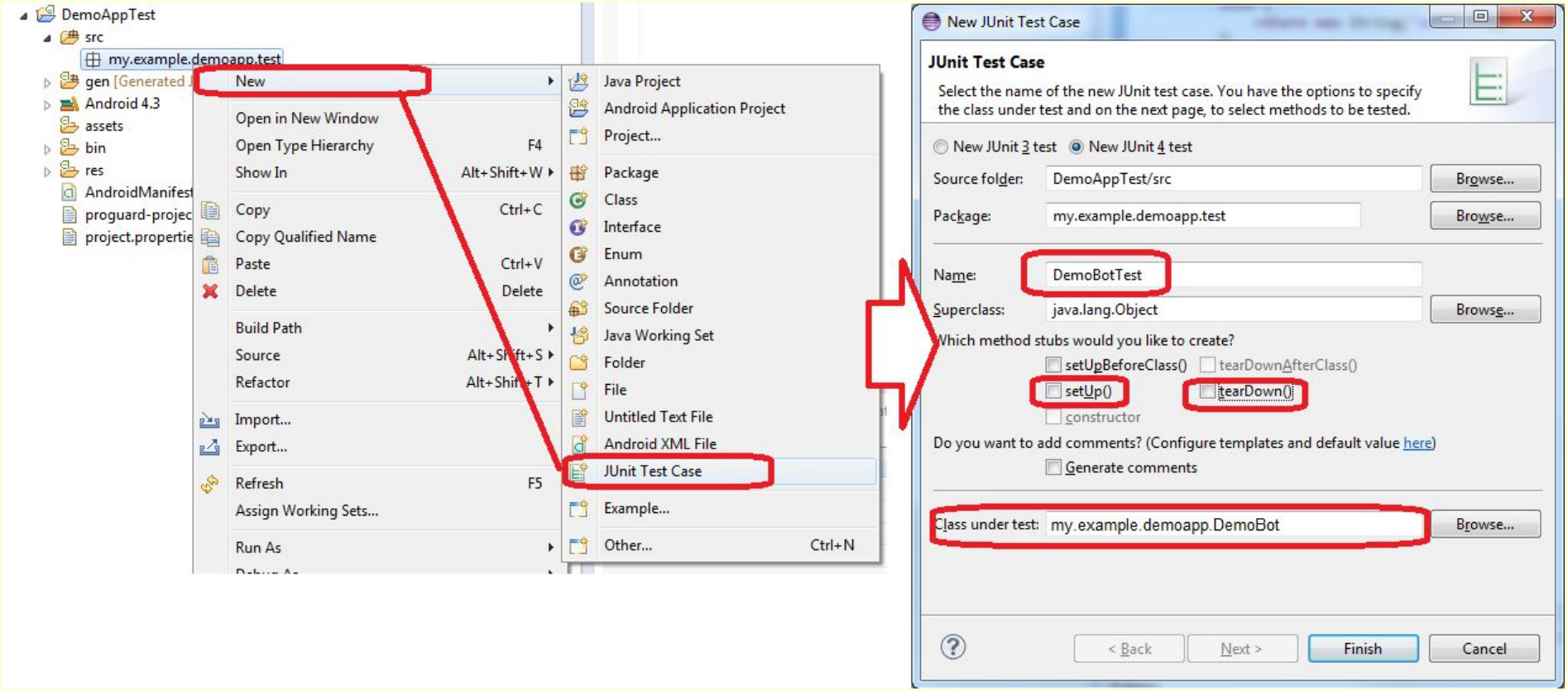
Проект с тестами: Java Project



Модули с тестами

- Классы, наследуемые от TestCase:
  - setUp() / tearDown()
  - public testXXX() – методы с тестами
    - наборы asserts – проверочных утверждений
- TestSuite – набор TestCases:
  - static suite() – фабричный метод для набора тестов
    - включает группу TestCase-ов и отдельных методов testXXX()
- Опционально – метод main() для запуска пакета тестов

Создание нового тестового модуля



Создание нового тестового модуля

DemoBot.java | DemoBotTest.java

```
package my.example.demoapp;

import java.util.Locale;

public class DemoBot {
    public String talkTo(String messageToBot)
    {
        if (messageToBot.toLowerCase(Locale.getDefault()).contains("Hello")) {
            return new String("Hi!");
        }
        else {
            return new String("Well...");
        }
    }
}
```

DemoBot.java | *DemoBotTest.java

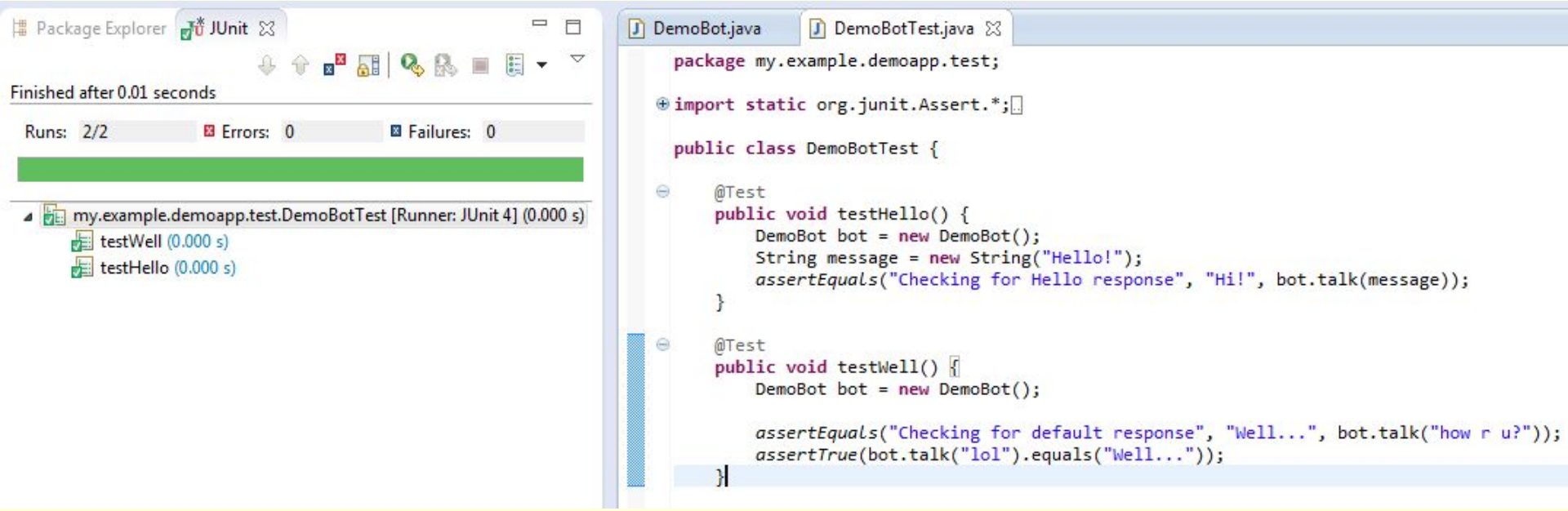
```
package my.example.demoapp.test;

import static org.junit.Assert.*;

public class DemoBotTest {

    @Test
    public void test() {
        fail("Not yet implemented");
    }

}
```

Виды проверочных утверждений

- assertXXX:
  - assertTrue(boolean test)
  - assertFalse(boolean test)
  - assertEquals(expected, actual)
  - assertSame(Object expected, Object actual)
  - assertNotSame(Object expected, Object actual)
  - assertNull(Object object)
  - assertNotNull(Object object)
- fail()
  - провал теста(генерирует AssertionFailedError)
- Все вышеуказанные методы в варианте с текстовым сообщением
  - напр., assertTrue(String message, boolean test)

Тестовые методы

- “Run As…->JUnit Test”
  - формируется отчёт
  - «красный» - есть не прошедшие тесты
  - «зелёный» - все тесты пройдены



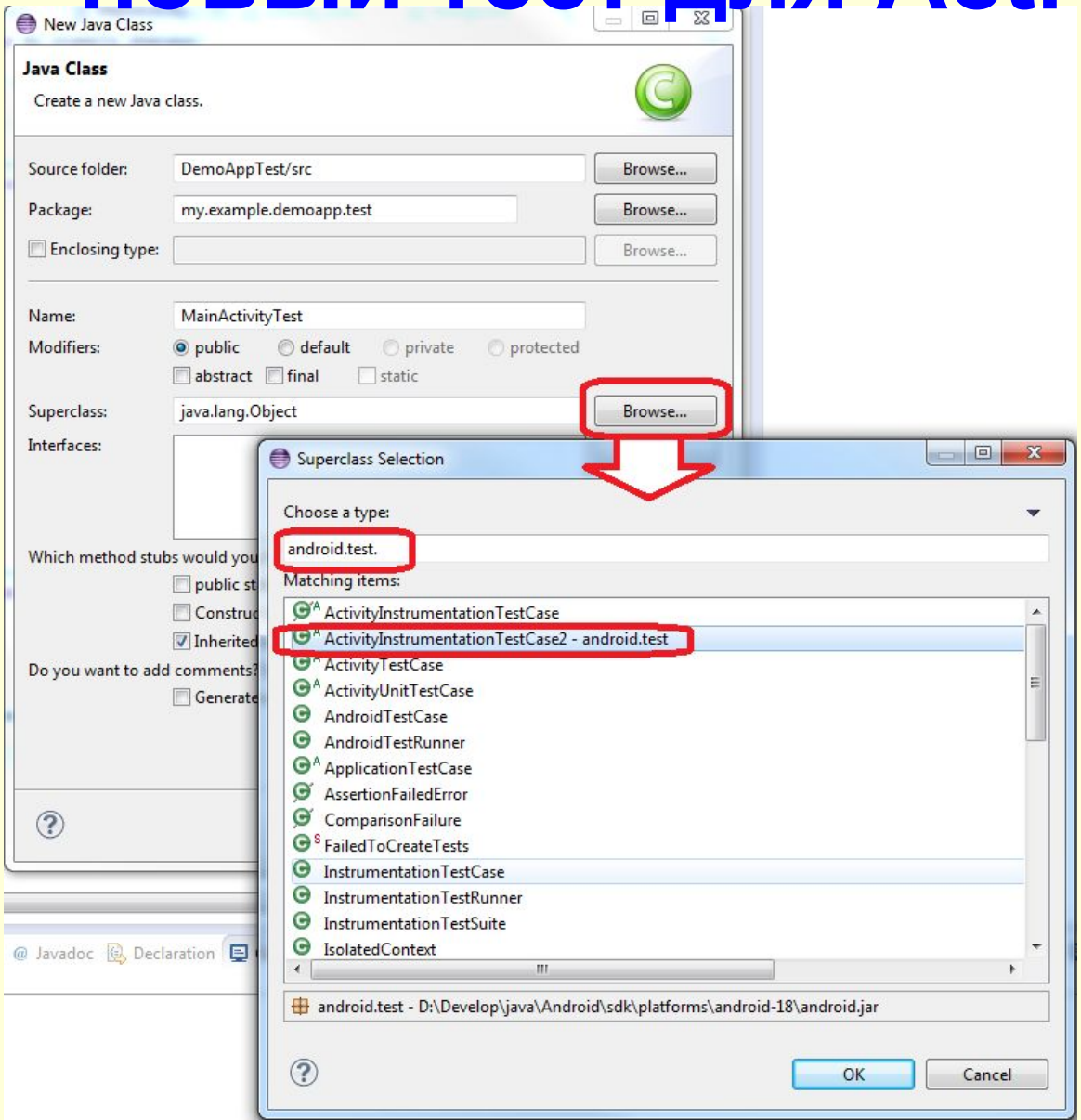
Fixtures

- Позволяют снизить дублирование кода
- Если для набора тестов нужны общие «предустановки», можно поместить их в метод setUp
  - в предыдущем примере, можно было бы сделать DemoBot полем класса DemoBotTest и инициализировать его в setUp()
    - чтобы к каждому тесту был создан «свежий» объект для тестирования

Тестирование GUI в Android средствами JUnit + android.test

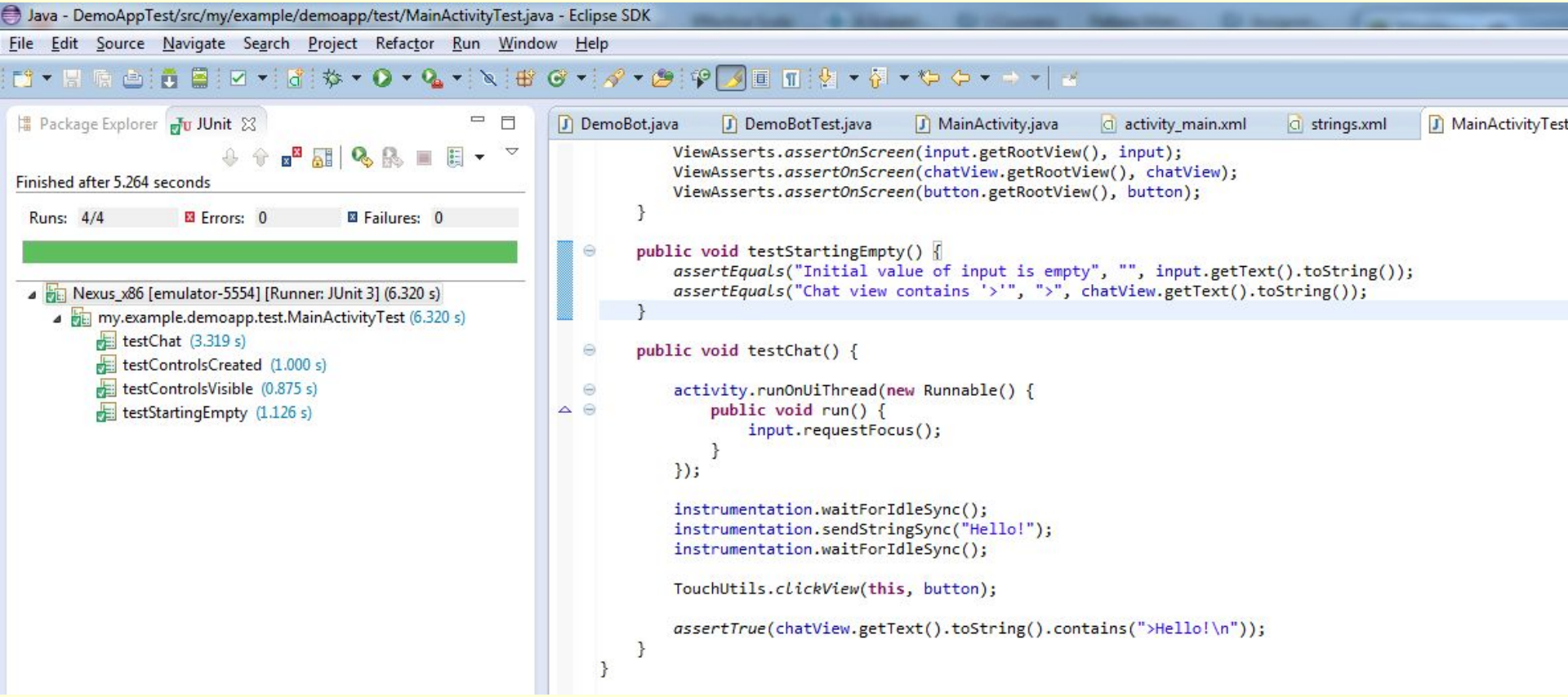
- android.test – фреймворк для написания специальных Android-тестов
  - GUI-тесты Android-компонент
    - большой набор assert-методов (см. документацию)
    - средства для изолированного тестирования GUI-классов (mock-и Android-компонент)
- Можно автоматизировать действия над интерфейсом и проверять:
  - видны ли элементы GUI, какие значения они содержат, как расположены и т.п.

android.test: новый тест для Activity



android.test: запуск тестов

"Run As" -> "Android JUnit Test"



Ссылки

- JUnit
  - документация: https://github.com/junit-team/junit/wiki
  - статья с примерами: http://habrahabr.ru/post/120101/
- Android Test:
  - Документация: http://developer.android.com/intl/ru/reference/android/test/package-summary.html
    - создание функциональных тестов: http://developer.android.com/intl/ru/training/activity-testing/activity-functional-testing.html
  - Тестирование Android-приложений: http://habrahabr.ru/post/113584/