

Best practices и особенности запуска unit-тестов

Даниил Пилипенко

Директор центра подбора IT-специалистов
SymbioWay

Skillbox

Best practices

- Тесты должны находиться отдельно от продакшн-кода — в папке `src/test`

Best practices

- Тесты должны находиться отдельно от продакшн-кода — в папке `src/test`
- Тесты должны размещаться зеркально основному коду — в таких же пакетах и классах

Best practices

- Тесты должны находиться отдельно от продакшн-кода — в папке `src/test`
- Тесты должны размещаться зеркально основному коду — в таких же пакетах и классах
- Имена классов с тестами могут оканчиваться на “Test” или “Tests” либо начинаться на “Test”

Best practices

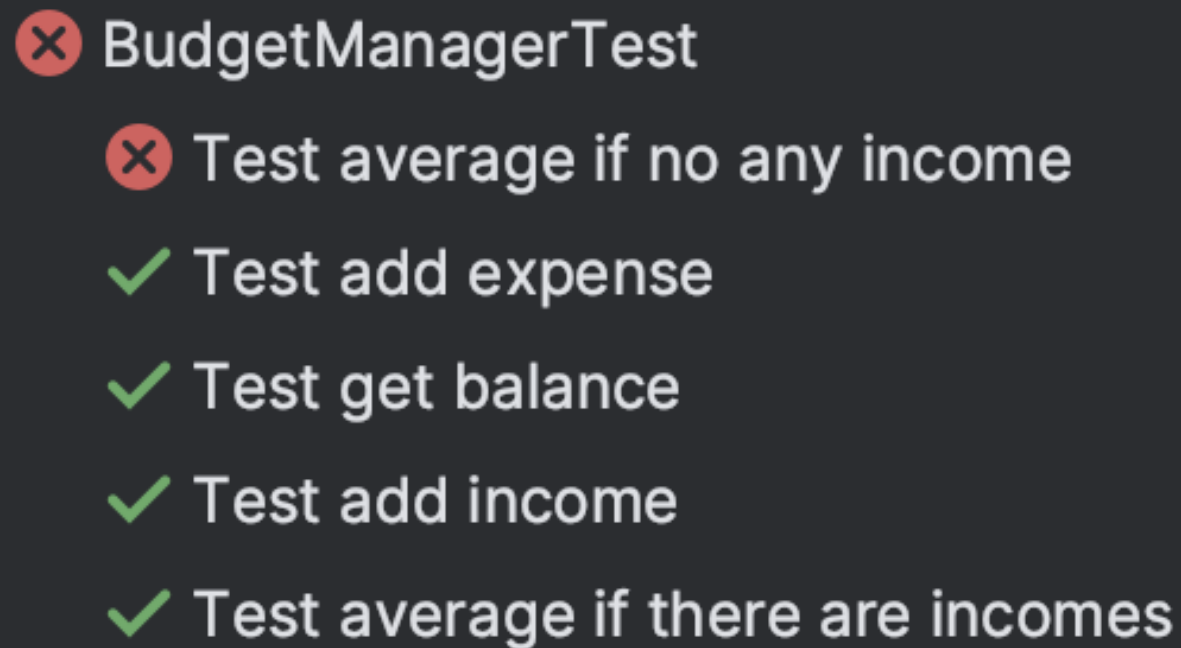
- Тесты должны находиться отдельно от продакшн-кода — в папке `src/test`
- Тесты должны размещаться зеркально основному коду — в таких же пакетах и классах
- Имена классов с тестами могут оканчиваться на “Test” или “Tests” либо начинаться на “Test”
- В тестовых классах можно создавать методы инициализации. Для их запуска перед каждым тестом нужно использовать аннотацию `@BeforeEach`

Best practices

- Тесты должны находиться отдельно от продакшн-кода — в папке `src/test`
- Тесты должны размещаться зеркально основному коду — в таких же пакетах и классах
- Имена классов с тестами могут оканчиваться на “Test” или “Tests” либо начинаться на “Test”
- В тестовых классах можно создавать методы инициализации. Для их запуска перед каждым тестом нужно использовать аннотацию `@BeforeEach`
- В тестовых классах создаются тестовые методы, которые нужно помечать аннотацией `@Test`

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`



- ✗ BudgetManagerTest
 - ✗ Test average if no any income
 - ✓ Test add expense
 - ✓ Test get balance
 - ✓ Test add income
 - ✓ Test average if there are incomes

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными

Примеры:

- `testMeasureObject`

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными

Примеры:

- `testMeasureObject`
- `givenZeroVolume_whenMeasureObject_thenZero`
- `givenVolume_whenMeasureObject_thenMetrics`

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка
- Для проверок нужно использовать подходящие `assert`-методы

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка
- Для проверок нужно использовать подходящие `assert`-методы

Примеры:

- `assertEquals(expected, actual)`
- `assertTrue(actual)`
- `assertNotNull(actual)`
- ...

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка
- Для проверок нужно использовать **подходящие** assert-методы

Не следует менять значения для проверки:

```
assertEquals("YES", actual);
```

```
assertTrue(actual.equals("YES"));
```

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка
- Для проверок нужно использовать подходящие `assert`-методы
- Для проверок нужно использовать максимально простые тест-кейсы и проверки, в которых актуальное значение сравнивается с хардкод-значением

```
assertEquals(5, actualValue);
```

Best practices

- Тестовые методы для удобства и понятности важно снабжать аннотацией `@DisplayName`
- Имена тестовых методов также должны быть понятными и единообразными
- Один тестовый метод (тест-кейс) — одна проверка
- Для проверок нужно использовать подходящие `assert`-методы
- Для проверок нужно использовать максимально простые тест-кейсы и проверки, в которых актуальное значение сравнивается с хардкод-значением
- Тесты нужно запускать после любых изменений в коде

Best practices и особенности запуска unit-тестов

Особенности запуска

Особенности запуска

- Тесты можно запускать из среды разработки на любых уровнях метода, класса, пакета или проекта

Особенности запуска

- Тесты можно запускать из среды разработки на любых уровнях метода, класса, пакета или проекта
- Для запуска тестов в Maven удобно использовать команду **`mvn test`**

Особенности запуска

- Тесты можно запускать из среды разработки на любых уровнях метода, класса, пакета или проекта
- Для запуска тестов в Maven удобно использовать команду **`mvn test`**
- Выполнить Maven-команду без запуска тестов можно при помощи ключа **`-DskipTests`**. Пример команды: **`mvn compile -DskipTests`**