



Лекция 7. Типы и уровни тестирования

Уровень тестирования

Уровень тестирования: определяет объект, который тестируется. Уровень тестирования связан с областью ответственности в проекте.

Примерами уровней тестирования могут служить

- компонентное тестирование,
- интеграционное тестирование,
- системное тестирование
- приёмочное тестирование.

Уровень тестирования

Для каждого уровня тестирования могут быть определены:

1. Цель
2. Объекты тестирования
3. Прослеживание связи с базисом тестирования (при наличии)
4. Критерии входа и выхода
5. Артефакты процесса тестирования, которые будет поставлять отдел тестирования - тестовые сценарии, протоколы тестирования, отчетность о результатах и другие
6. Тестовые методики
7. Измерения и метрики
8. Инструментарий

Уровни тестирования

Процесс тестирования включает в себя следующие уровни:



Компонентное тестирование

Компонентное тестирование: Тестирование отдельных компонентов программного обеспечения



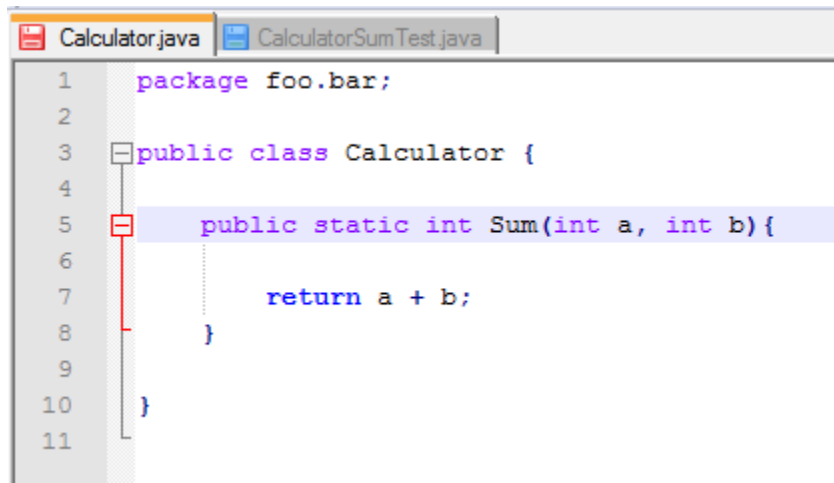
Также известно как **М**одульное тестирование

Компонентное тестирование: общий обзор

- Выполняется самим разработчиком (иногда модульное тестирование доверяется другому разработчику, не автору кода, для повышения уровня независимости)
- Тестирование функциональных и нефункциональных характеристик программы
- Могут быть использованы эмуляторы (заглушки и драйвера)

Компонентное тестирование

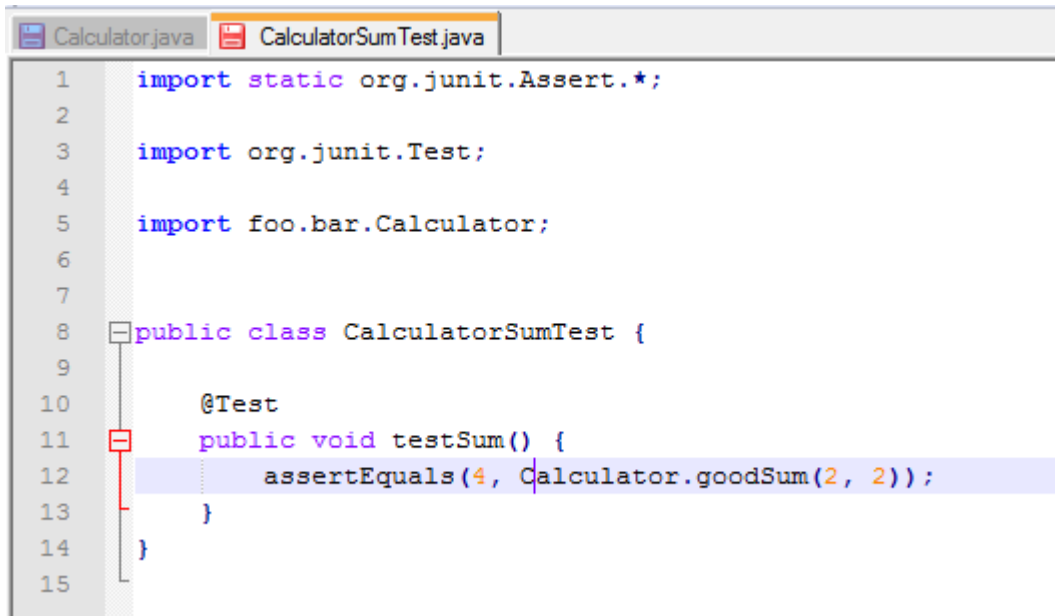
■ Пример кода



```
1 package foo.bar;
2
3 public class Calculator {
4
5     public static int Sum(int a, int b) {
6
7         return a + b;
8     }
9
10 }
11
```

Компонентное тестирование

■ Модульный тест



The screenshot shows a Java IDE with two tabs: 'Calculator.java' and 'CalculatorSumTest.java'. The 'CalculatorSumTest.java' tab is active, displaying the following code:

```
1  import static org.junit.Assert.*;
2
3  import org.junit.Test;
4
5  import foo.bar.Calculator;
6
7
8  public class CalculatorSumTest {
9
10     @Test
11     public void testSum() {
12         assertEquals(4, Calculator.goodSum(2, 2));
13     }
14 }
15
```

- Далее эту программу запускаем, таким образом автоматически тестируют код.

Компонентное тестирование

- **Цель**
 - Изолировать отдельные части программы и показать, что по отдельности все части работают.
- **Объекты тестирования**
 - Компоненты
 - Программы
 - Модули БД
- **Базис тестирования**
 - Требования к компонентам
 - Детальный дизайн
 - Код

Компонентное тестирование

■ Критерии входа

- Бизнес- и функциональные требования выработаны и утверждены.
- Разработка компонентов закончена.
- Среда разработки стабильна.
- Документация по тестовым сценариям модульных тестов составлена.

■ Критерии выхода

- Все тестовые сценарии модульных тестов исполнены. Тестовые результаты доступны.
- Обнаруженные дефекты исправлены и закрыты.
- Проверка кода завершена.
- Все обнаруженные серьезные дефекты закрыты.

Компонентное тестирование

■ Отчетность

- Как правило, дефекты устраняются по мере обнаружения, без составления отчетов.

■ Тестовые методики

- Тестирование операторов
- Тестирование ветвей
- Тестирование условий
- Тестирование путей

■ Метрики и измерения

- Покрытие операторов
- Покрытие ветвей
- Покрытие путей

Компонентное тестирование

- **И**нструментарий
 - Инструмент отладки
 - Интегрированная среда модульного тестирования
 - Junit
 - Nunit
 - Другие

Компонентное тестирование

Преимущества

- Возможность протестировать часть программы, не ожидая готовности остальных частей
- Раннее обнаружение дефектов
- Программисты обнаруживают и мгновенно исправляют проблемы. Упрощенная отладка
- Лучшее структурное покрытие кода
- Модульное тестирование экономичнее других этапов тестирования
- Упрощенная интеграция

Компонентное тестирование

Недостатки

- Время от времени требуется реализовывать *заглушки* и *драйверы*
- Модульное тестирование основано, в первую очередь, на написанном коде. Поэтому, если что-то было пропущено, модульное тестирование этого не покажет

Тестирование интеграции компонентов

Тестирование интеграции компонентов:

Тестирование, выполняемое для выявления дефектов в интерфейсах и взаимодействии между интегрированными компонентами.



Тестирование интеграции компонентов: общий обзор

- Как правило, следует за компонентным тестированием
- Выполняется разработчиками или тестировщиками, специализирующихся на интеграционном тестировании (редкая квалификация)
- Тестирование функциональных и нефункциональных характеристик программы

Тестирование интеграции компонентов

- **Цель**

- Удостовериться, что взаимодействие двух или более компонентов дает результат, отвечающий функциональным требованиям
- Обнаружить проблемы интерфейса

- **Объекты тестирования**

- Подсистемы
- Инфраструктура
- Интерфейсы

Тестирование интеграции компонентов

■ Базис тестирования

- Проект программы и системы
- Архитектура
- Технологический процесс (workflow)
- Сценарии использования

■ Критерии входа

- Модули для интеграционного тестирования закончены
- Компонентное тестирование закончено
- Проблемы, обнаруженные в компонентном тестировании, исправлены и закрыты
- Сценарии интеграционного тестирования закончены
- Среда интеграционного тестирования готова

Тестирование интеграции компонентов

■ Критерии выхода

- Дефекты, обнаруженные во время интеграционного тестирования, исправлены и закрыты
- Все тестовые сценарии исполнены; для каждого сценария есть результаты тестирования

■ Методы интеграционного тестирования

- «Большой взрыв» (“Big Bang”)
- «Сверху вниз» (“Top down”)
- «Снизу вверх» (“Bottom up”)
- *Методы подробно разбираются в тренинге «SQA-028 Основы тест-дизайна»*

Тестирование интеграции компонентов

Преимущества

- Большая стабильность по сравнению с тестированием графического пользовательского интерфейса
- Положительно влияет на внутренний дизайн программы
- Ранняя и более легкая локализация дефектов интерфейса на стадии системного тестирования

Недостатки

Тестировщик должен читать код, а временами и писать его

Системное тестирование

Системное тестирование: Процесс тестирования системы в целом с целью проверки того, что она соответствует установленным требованиям



Системное тестирование: общий обзор

- Выполняется тестировщиками
- Тестирование функциональных и нефункциональных характеристик программы
- Системное тестирование является разновидностью тестирования методом черного ящика, а, следовательно, не требует знания внутренней структуры кода или логики
- Включает тестирование взаимодействия с операционной системой и системными ресурсами



Системное тестирование

- **Цель**
 - Протестировать законченную, интегрированную систему, чтобы оценить ее соответствие указанным требованиям (функциональным/нефункциональным)
- **Объекты тестирования**
 - Система в целом
- **Базис тестирования**
 - Функциональная спецификация (FRS)
 - Спецификация системных требований к ПО (SRS)
 - Сценарии использования
 - Отчеты об анализе степени риска



Системное тестирование

■ Критерии входа

- Модульное и интеграционное тестирование всех модулей закончено.
- Окружение для системного тестирования готово.
- Спецификации продукта закончены и утверждены.
- Сценарии системного тестирования отражены в документах.
- Пользовательский интерфейс и тестируемый функционал заморожены.

■ Критерии выхода

- Программа отвечает всем требованиям и обладает требуемым функционалом.
- Дефекты, обнаруженные во время системного тестирования, исправлены и закрыты.
- Все сценарии системного тестирования исполнены, а результаты доступны.



Системное тестирование

■ Техники тест-дизайна

- Эквивалентное разбиение
- Анализ граничных значений
- Тестирование таблицы решений
- Тестирование всех пар (pairwise)
- Тестирование состояний и переходов
- Тестирование по сценариям использования

■ Измерения и метрики

- Покрытие требований
- Покрытие классов эквивалентности
- Покрытие граничных значений



Системное тестирование

■ Отчетность

- Данные по обнаруженным дефектам, как правило, заносятся в отчет и в систему управления дефектами

■ Инструментарий

- Тестовые компараторы
- Инструменты захвата/воспроизведения
- Инструменты тестирования защищенности
- Инструменты тестирования производительности
- ...

Приемочное тестирование

Приемочное тестирование - формальное испытание системы, проводимое с целью определения соответствия реализованных требований, бизнес процессов, потребностей пользователя приемочным критериям. На основании результатов приемочного тестирования пользователь, заказчик или другое уполномоченное лицо принимает решение о приемке системы в эксплуатацию



Приемочное тестирование: общий обзор

- Выполняется заказчиком или пользователем системы
- Поиск дефектов не является главной целью
- Пользовательское приемочное тестирование проверяет готовность системы для использования; Эксплуатационное тестирование проверяет насколько система пригодна для эксплуатации в конкретном операционном окружении
- Альфа тестирование выполняется в стенах компании, которая разрабатывает программный продукт. Бета тестирования выполняется заказчиком/пользователем на его оборудовании

Классификация тестирования

- С исполнением и без исполнения кода:
статическое / динамическое
- Различные знания о структуре кода:
черный ящик / серый ящик / белый ящик
- По свойствам тестируемого объекта:
функциональность, производительность,
совместимость, надежность, удобство...
- По изменениям:
регрессионное тестирование, подтверждающее
тестирование
- По типу прогона тестов:
ручное и автоматическое



С исполнением и без исполнения кода...

Статическое тестирование: Тестирование компонента или системы на уровне спецификации или реализации без исполнения кода программного продукта, например, рецензирование или статический анализ.

Динамическое тестирование: Тестирование, проводимое во время выполнения программного обеспечения, компонента или системы.



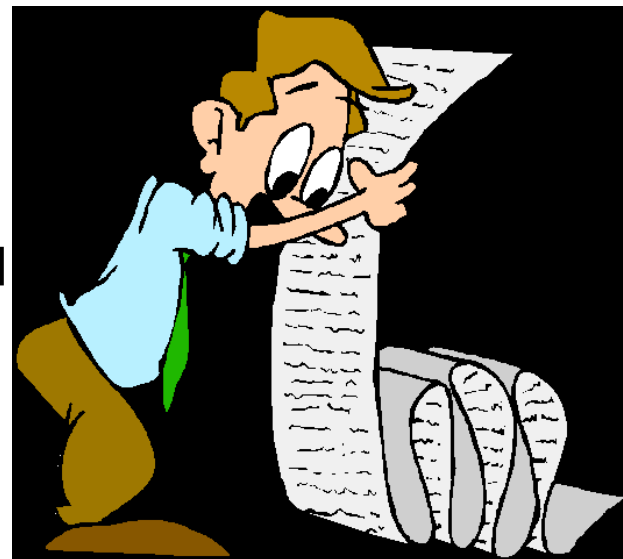
Статическое тестирование

При статическом тестировании код исследуется вручную (рецензирование) или с помощью автоматизированных средств анализа (статистический анализ) **без исполнения кода**.

Объекты тестирования:

- Код
- Документация с требованиями
- Сценарии использования
- Руководства

.. и прочая проектная документация



Статическое тестирование

Преимущества рецензирования:

- раннее обнаружение и исправление дефектов
- улучшение продуктивности разработки
- уменьшение времени разработки
- уменьшение времени и стоимости тестирования
- сокращение стоимости жизненного цикла
- уменьшение числа дефектов и улучшение коммуникаций
- могут быть найдены упущения в требованиях



Динамическое тестирование

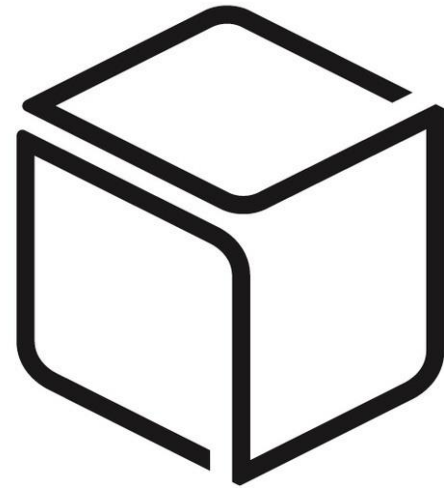
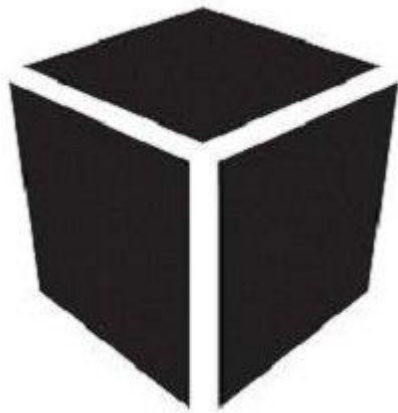
Цель статического тестирования – **поиск дефектов** в продукте, в то время как цель динамического тестирования, обнаружение **отказов системы**. Только динамическое тестирование дает представление о поведении программы и позволяет выявить различия между ожидаемым и фактическим поведением.

Объекты тестирования:

- Модуль
- Интерфейс
- Система



Различные знания о структуре кода...



Тестирование методом черного ящика

Тестирование методом черного ящика:

Тестирование, функциональное или нефункциональное, без знания внутренней структуры компонента или системы.



Тестирование методом серого ящика

Тестирование методом серого ящика:

Сочетает в себе тестирование методом черного и белого ящика.

- Например, продукт тестируется методом черного ящика, но тестовые сценарии разрабатываются с учетом знаний о внутренней структуре продукта.



Тестирование методом белого ящика

Тестирование методом белого ящика:

Тестирование, основанное на анализе внутренней структуры компонента или системы.

Синонимы:

- *тестирование на основе структуры*
- *структурное тестирование*
- *тестирование прозрачного ящика*
- *тестирование методом прозрачного ящика –*

По свойствам тестируемого объекта...

■ Функциональное тестирование

- Тестирование установки (инсталляции)
- Тестирование графического пользовательского интерфейса

■ Нефункциональное тестирование

- Тестирование производительности, нагрузочное тестирование, стрессовое тестирование
- Тестирование обеспеченности технической поддержкой
- Тестирование локализуемости
- Тестирование практичности
- Тестирование защищенности
- ...

Функциональное тестирование

Функциональное тестирование:

Тестирование, основанное на анализе спецификации функциональности компонента или системы.

Функциональное тестирование включает в себя:

- Тестирование настройки и лицензирования
- Тестирование графического пользовательского интерфейса
- ...



Тестирование установки и лицензирования

Тестирование установки: Вид тестирования, ориентированный на то, что требуется пользователям для успешной установки, настройки и регистрации программного продукта. Процесс тестирования может включать полный и частичный процесс установки/удаления, а так же обновления программы

Тестирование целостности данных

Тестирование целостности данных: Вид тестирования, главной целью которого является проверка целостности данных после различных транзакций (ввод/выбор/обновление). Как правило, целостность данных проверяется в тестированиях методом белого и черного ящика

Тестирование защищенности

Тестирование защищенности: Тестирование с целью оценить защищенность программного продукта

Объекты тестирования:

- пароли
- шифрование
- аппаратные устройства доступа
- уровни доступа к информации
- авторизация
- скрытые каналы
- безопасность на физическом уровне

Тестирование графического пользовательского интерфейса

Тестирование графического
пользовательского интерфейса: Тестирование
графического пользовательского интерфейса
продукта с целью удостовериться в том, что он
отвечает спецификациям

Нефункциональное тестирование

Нефункциональное тестирование: Тестирование атрибутов компонента или системы, не относящихся к функциональности, то есть надежность, эффективность, практичность, сопровождаемость и переносимость.



Тестирование производительности

Тестирование производительности: Процесс тестирования с целью определить производительность программного продукта



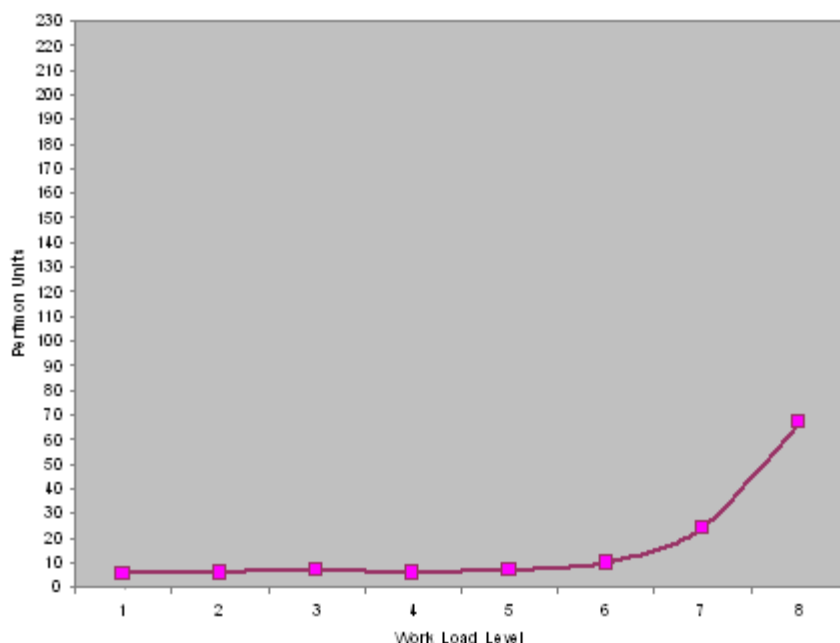
Тестирование производительности: в инженерии программного обеспечения тестирование, которое проводится с целью определения, как быстро работает система или её часть под определённой нагрузкой. Также может служить для проверки и подтверждения других атрибутов качества системы, таких как масштабируемость, надёжность и потребление ресурсов. <http://ru.wikipedia.org/>



Тренинг "SQA-033 Основы тестирования производительности"

Нагрузочное тестирование

Нагрузочное тестирование: Тип тестирования производительности, проводимый с целью оценки поведения компонента или системы при возрастающей нагрузке, например количестве параллельных пользователей и/или операций, а также определения какую нагрузку может выдержать компонент или система.



Стрессовое тестирование

Стрессовое тестирование: Вид тестирования производительности, оценивающий систему или компонент на граничных значениях рабочих нагрузок или за их пределами, или же в состоянии ограниченных ресурсов, таких как память или доступ к серверу



Тестирование удобства использования

Тестирование удобства использования:

Тестирование с целью определения степени понятности, легкости в изучении и использовании, привлекательности программного продукта для пользователя при условии использования в заданных условиях эксплуатации.



Тестирование по изменениям...

Подтверждающее тестирование: Тестирование, во время которого исполняются тестовые сценарии, выявившие ошибки во время последнего запуска, для подтверждения успешности исправления этих ошибок.

Регрессионное тестирование: Тестирование уже протестированной программы, проводящееся после модификации для уверенности в том, что процесс модификации не внес или не активизировал ошибки в областях, не подвергавшихся изменениям. Проводится после изменений в коде программного продукта или его окружении

По типу прогона тестов..

Ручное тестирование: Процесс ручного тестирования продукта. Тестировщик играет роль конечного пользователя, используя максимальное количество функций программы, чтобы удостовериться в их корректной работе.

Автоматизированное тестирование: Использование программного обеспечения (помимо тестируемого ПО) для контроля выполнения тестов, сравнения полученных результатов с эталонными, установки предусловий тестов и других функций контроля тестирования и организации отчетов.