

# Теория тестирования, часть №1 (подход, технологии, уровни)

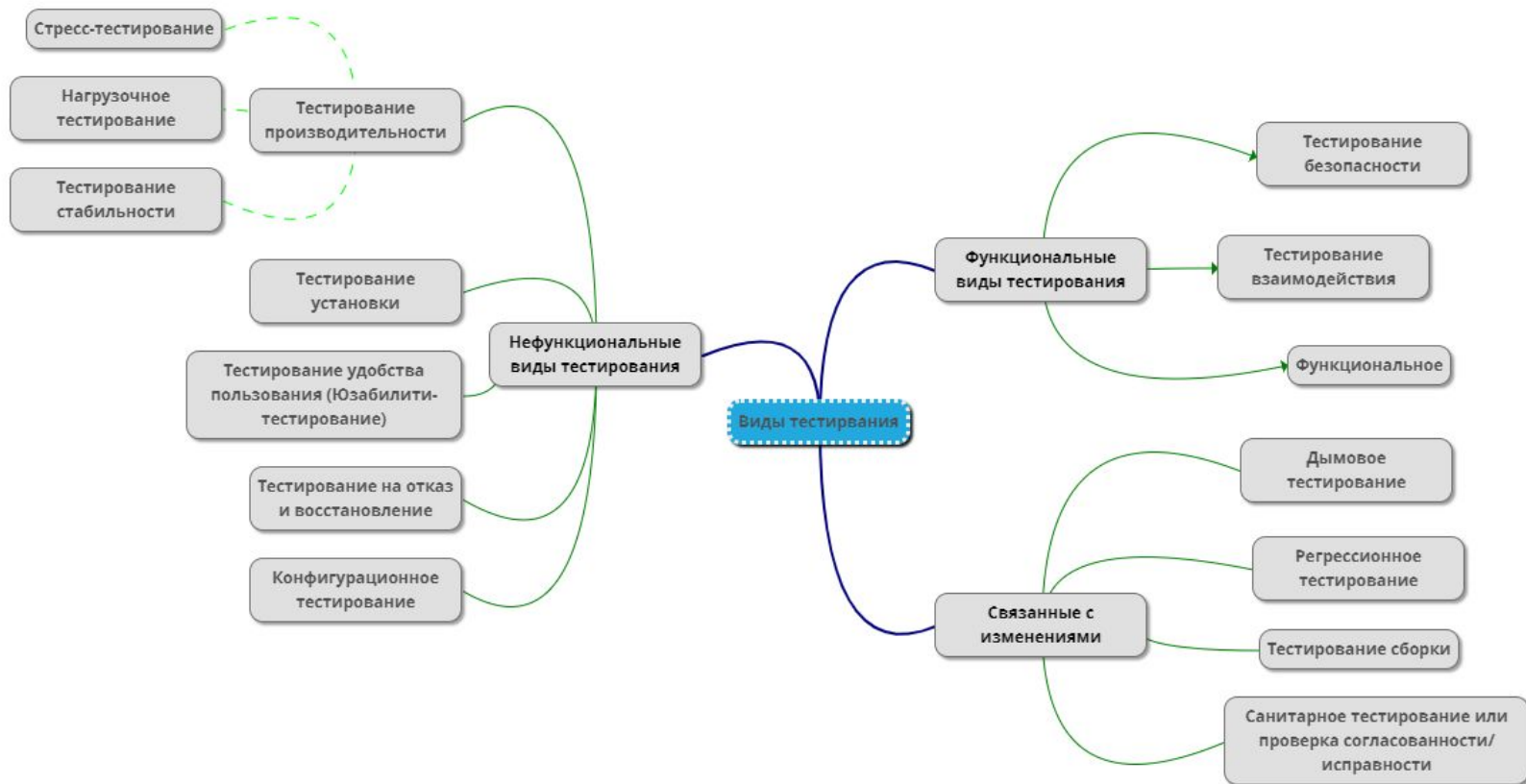
Урок 6




# План

1. Виды тестирования.
2. Уровни тестирования
3. Технологии тестирования.
4. Подходы к тестированию.
5. Инструменты для снятия скриншотов.

# Виды тестирования





Различие задач тестирования приводит к необходимости использовать весьма разнообразные типы (виды) тестирования. Принято подразделять тестирование на виды по следующим категориям:

- по объектам (элементам) тестирования, часто разделение на виды тестов по данному критерию называют разделением тестирования на уровни;
- по глубине тестирования, то есть разделение тестовых испытаний на типы проводится в зависимости от количества времени и объема тестируемых компонент программного продукта.

Тем не менее, основная классификация тестов на виды производится в соответствии с традиционными показателями качества, которые проверяются с их помощью.

# Виды тестирования.

Все виды тестирования программного обеспечения, в зависимости от преследуемых целей, можно условно разделить на следующие группы:

1. Функциональные
2. Нефункциональные
3. Связанные с изменениями



# Функциональные виды тестирования.

Функциональные тесты базируются на функциях и особенностях, а также взаимодействии с другими системами, и могут быть представлены на всех уровнях тестирования: компонентном или модульном (Component/Unit testing), интеграционном (Integration testing), системном (System testing). Функциональные виды тестирования рассматривают внешнее поведение системы. Самыми распространенными видами функциональных тестов являются:

- Функциональное тестирование (Functional testing)
- Тестирование безопасности (Security and Access Control Testing)
- Тестирование взаимодействия (Interoperability Testing)

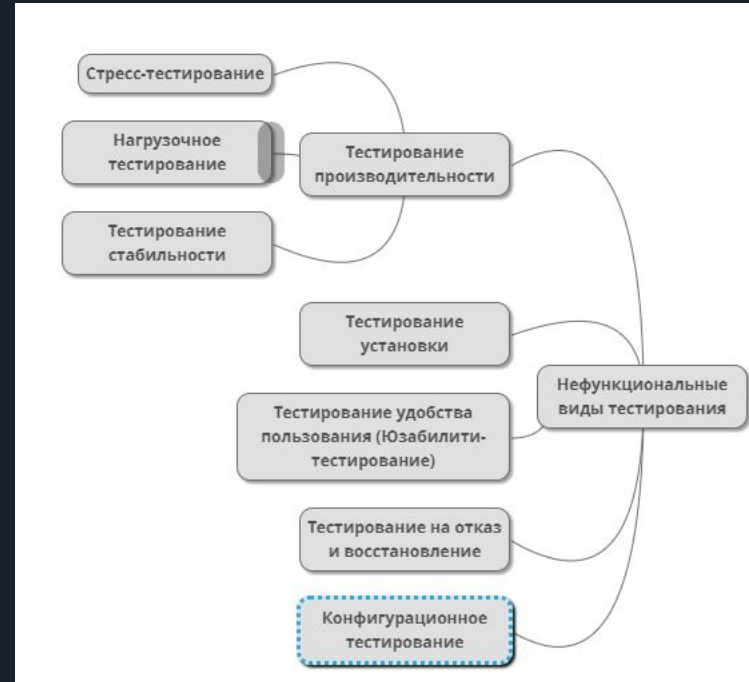


# Нефункциональные виды тестирования.

Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, которые могут быть измерены различными величинами. Это тестирование того, "Как" система работает.

Основными видами нефункциональных тестов являются:

- Все виды тестирования производительности: нагрузочное тестирование (Performance and Load Testing) стрессовое тестирование (Stress Testing) тестирование стабильности или надежности (Stability / Reliability Testing) объемное тестирование (Volume Testing)
- Тестирование установки (Installation testing)
- Тестирование удобства пользования (Usability Testing)
- Тестирование на отказ и восстановление (Failover and



# Виды тестирования связанные с изменениями

После проведения необходимых изменений, таких как исправление дефектов, программное обеспечение должно быть протестировано заново, для подтверждения того факта, что проблема была решена.

Ниже перечислены виды тестирования, которые необходимо проводить после установки программного обеспечения, для подтверждения работоспособности приложения или правильности

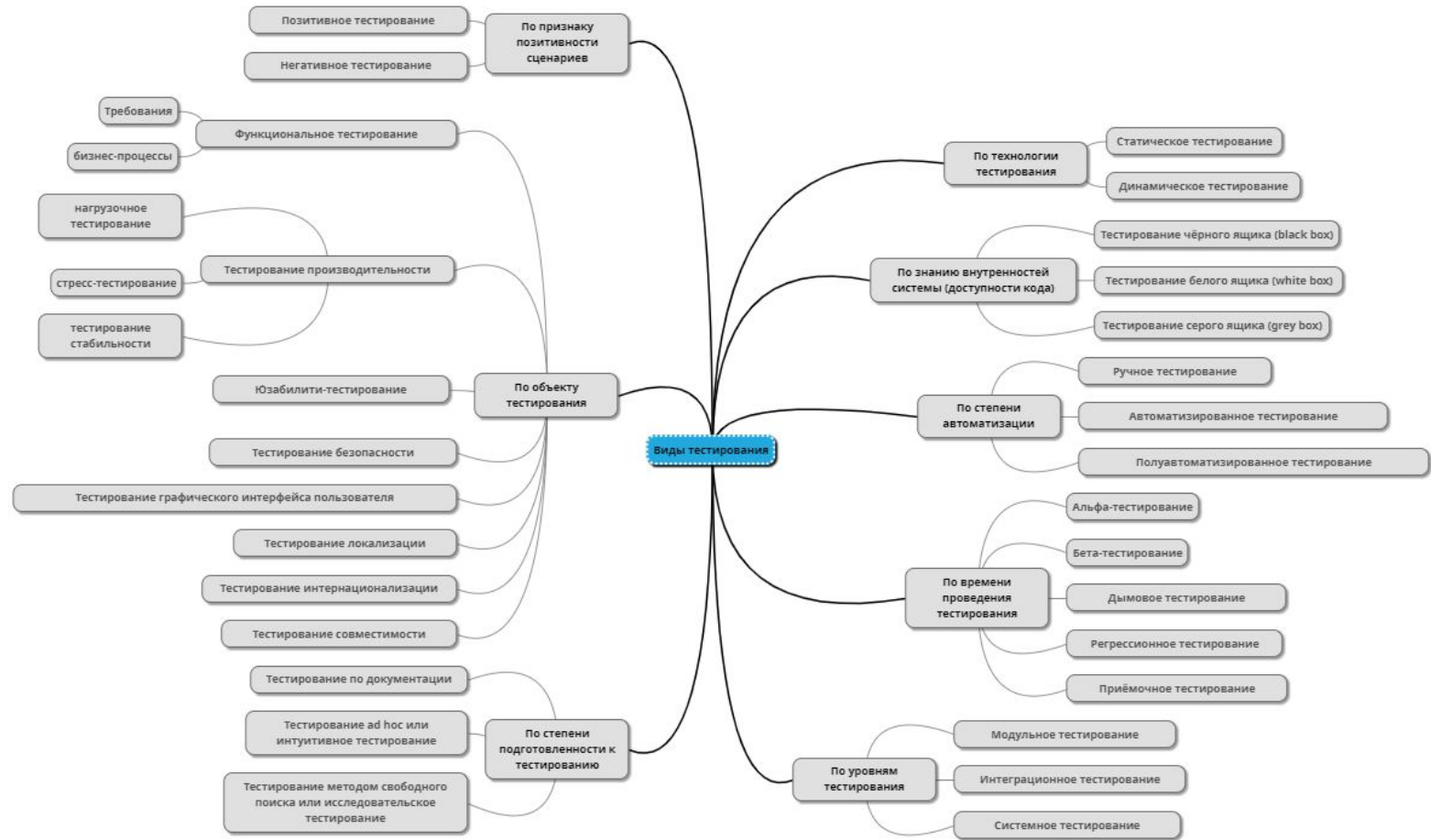
осуществленного исправления дефекта:

- Дымовое тестирование (Smoke Testing)
- Регрессионное тестирование (Regression Testing)
- Тестирование сборки (Build Verification Test)
- Санитарное тестирование или проверка





Производить классификацию видов тестирования, также принято по следующим признакам:



# По знанию внутренностей системы (доступности кода):

- Тестирование чёрного ящика (black box).

Тестирование проводится без доступа к исходному коду;

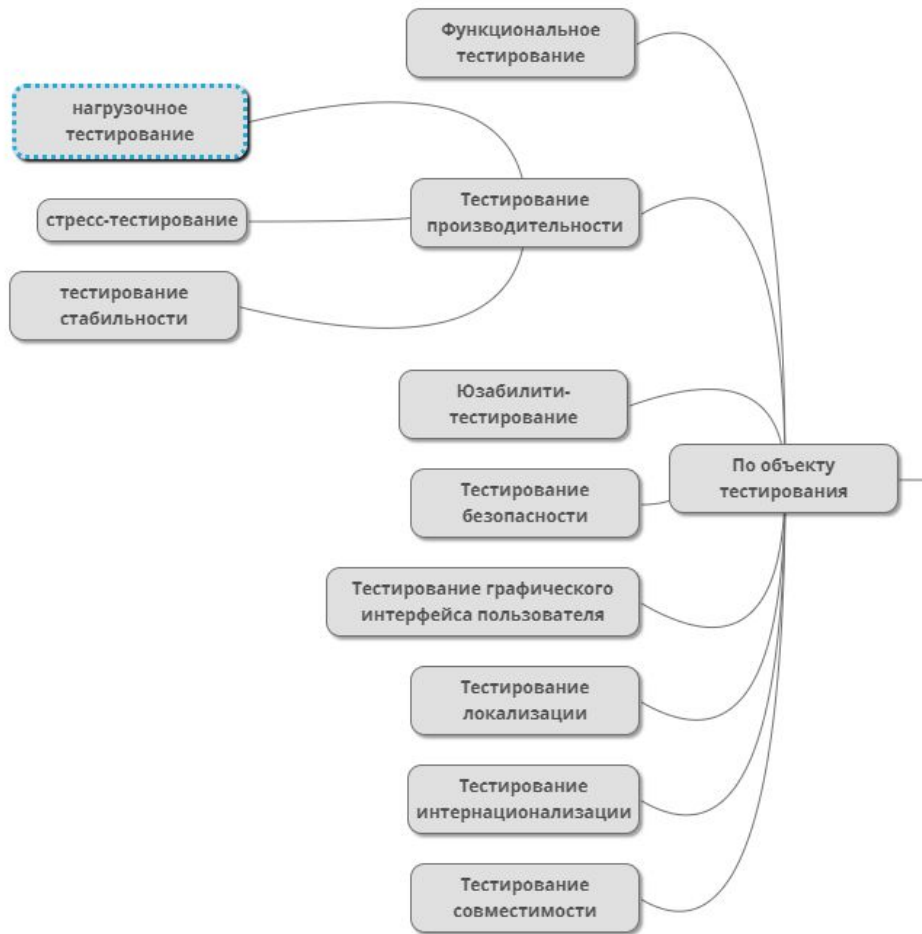
- Тестирование белого ящика (white box).

Тестирование проводится с доступом к исходному коду и с возможностью модификации кода;

- Тестирование серого ящика (grey box). Представляет собой объединение двух выше перечисленных видов тестирования. Разработчик тестов имеет доступ к исходному коду, но при непосредственном выполнении тестов доступ к коду, как правило, не требуется.



# По объекту тестирования:





# По объекту тестирования:

- Функциональное тестирование (functional testing). Проверка функций и характеристик разрабатываемого ПО на основе проектной документации;
- Тестирование производительности (performance testing). Проверка работоспособности системы под нагрузкой. Может служить для проверки и подтверждения других атрибутов качества системы, таких как масштабируемость, надёжность и потребление ресурсов. Выделяются следующие подвиды: нагрузочное тестирование (load testing), стресс-тестирование (stress testing), тестирование стабильности (stability testing);
- Юзабилити-тестирование (usability testing). Цель данного вида тестирования заключается в определении степени удобства и практичности пользовательского интерфейса;
- Тестирование безопасности (security testing). Проверка надежности системы от возможных рисков и угроз (потеря, конфиденциальность, целостность и доступность данных);
- Тестирование графического интерфейса пользователя (GUI) - это тип тестирования программного обеспечения, который проверяет графический интерфейс пользователя тестируемого приложения;
- Тестирование локализации (localization testing). Корректность работы отдельных компонентов системы (форматы дат, единицы измерения, не меняется часовой пояс при переезде и т.д.);
- Тестирование интернационализации (internationalization). Позволяет убедиться в поддержке культурных особенностей других стран (главным образом, в языковой поддержке).
- Тестирование совместимости (compatibility testing) Проверка возможности приложения взаимодействовать с различными программными продуктами, операционными системами и окружением.

# По степени автоматизации:

- Ручное тестирование (manual testing). Тестирование проводится без инструментов автоматизации;
- Автоматизированное тестирование (automated testing). Тестирование на всех уровнях выполняется с использованием средств автоматизации;
- Полуавтоматизированное тестирование (semiautomated testing). Предполагается, что для определенных целей применяется автоматизация (автоматизация развертки окружения, автоматизация функционального тестирования и т.д.)



# По времени проведения тестирования:

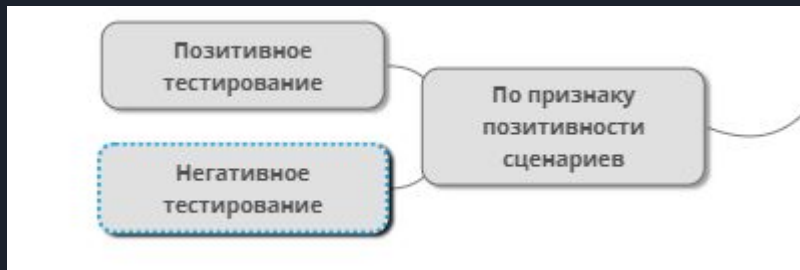
- Альфа-тестирование. Внутреннее тестирование (имитация реальной работы с системой штатными разработчиками).
- Бета-тестирование. Внешнее пробное использование с привлечением отобранных пользователей.
- Дымовое тестирование (smoke testing). Минимальный набор тестов на явные ошибки;
- Тестирование новой функциональности (new feature testing). Тестируются новые функции системы;
- Регрессионное тестирование (regression testing). Проверка изменений сделанных в системе для подтверждения того факта, что существующая ранее функциональность работает как и прежде;
- Приёмочное тестирование (acceptance testing). Целью приемочного тестирования является оценка готовности системы для его выпуска на рынок или передачи клиенту. Может включать в себя альфа-тестирование (alpha testing) и бета-тестирование (beta testing).



# По признаку позитивности сценариев:

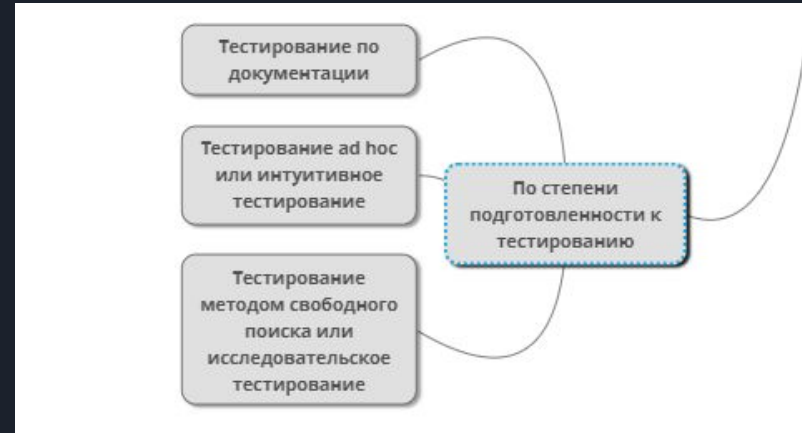
Позитивное тестирование (positive testing). Проверка позитивных (правильных) пользовательских сценариев. На вход подается разрешенные (ожидаемые) данные;

Негативное тестирование (negative testing). Проверка реакции системы на ввод негативных (не разрешенных) данных.



# По степени подготовленности к тестированию:

- Тестирование по документации (formal testing).  
Тестирование приложения проводится по заранее подготовленным данным (тест-кейсы, чек-листы, читлисты, спецификация и т.д.)
- Тестирование ad hoc или интуитивное тестирование (ad hoc testing) — тестирование проводится при полном отсутствии документации, без плана и цели;
- Тестирование методом свободного поиска или исследовательское тестирование (exploratory testing) — предполагается наличие минимально необходимой для тестирования документации, либо тестирование без документации.







# Функциональное тестирование-


это вид тестирования, при котором выявляется некорректная /неправильная работа функционала программы. Этот вид тестирования занимает 90% времени отведенного на тестирование.

Функциональное тестирование предполагает проверку функциональных требований: логики и бизнес-правил приложения или системы.

Полноценное системное/функциональное тестирование является самым трудоемким процессом и может занимать до 80% всего бюджета проекта по тестированию.(Ф.Брукс)

Тестирование функциональности может проводиться в двух аспектах:

- требования
- бизнес-процессы



Тестирование в перспективе «требования» использует спецификацию функциональных требований к системе как основу для дизайна тестовых случаев (о которых мы поговорим в последующих лекциях). В этом случае необходимо сделать список того, что будет тестироваться, а что нет, приоритезировать требования на основе рисков (если это не сделано в документе с требованиями), а на основе этого приоритезировать тестовые сценарии. Это позволит сфокусироваться и не упустить при тестировании наиболее важный функционал.

Тестирование в перспективе «бизнес-процессы» использует знание бизнеспроцессов, которые описывают сценарии ежедневного использования системы. В этой перспективе тестовые сценарии, как правило, основываются на случаях использования системы.


Необходимо помнить о том, что полностью протестировать всё невозможно, необходимо постоянно отслеживать приоритетность задач на текущий момент



# Уровни тестирования.

Тестирование на разных уровнях производится на протяжении всего жизненного цикла разработки и сопровождения программного обеспечения. Уровень тестирования определяет то, над чем производятся тесты: над отдельным модулем, группой модулей или системой, в целом.

Проведение тестирования на всех уровнях системы - это залог успешной реализации и сдачи проекта




Модульное тестирование ( Unit testing) - тестируется минимально возможный для тестирования компонент, например, отдельный класс или функция. Часто модульное тестирование осуществляется разработчиками программного обеспечения.

Интеграционное тестирование ( Integration testing) - тестируются интерфейсы между компонентами, подсистемами или системами. При наличии резерва времени на данной стадии тестирование ведётся итерационно, с постепенным подключением последующих подсистем.

Системное тестирование ( System testing) - тестируется интегрированная система на её соответствие требованиям.





Для каждого уровня тестирования могут использоваться различные виды тестирования, для каждого из которых, в свою очередь, могут использоваться различные типы тестовых испытаний. Приемочное тестирование (Acceptance Testing)- это формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится, с целью определения удовлетворяет ли система приемочным критериям и вынесения решения заказчиком или другим уполномоченным лицом принимается приложение или нет. Приемочное тестирование выполняется на основании набора типичных тестовых случаев и сценариев, разработанных на основании требований к данному приложению.

Решение о проведении приемочного тестирования принимается, когда:

- продукт достиг необходимого уровня качества;
- заказчик ознакомлен с Планом Приемочных Работ (Product Acceptance Plan) или другим документом, где описан набор действий, связанных с проведением приемочного тестирования, дата проведения, ответственные.

Фаза приемочного тестирования длится до тех пор, пока заказчик не выносит решение об отправлении приложения на доработку или выдаче приложения.



# Технологии тестирования.

Технологий тестирования существует целое множество. Условно их можно отнести к статическим или к динамическим.

Статическое тестирование – это процесс, который обычно ассоциируют с анализом ПО. Статическим тестированием пользуются для верификации практически любого артефакта разработки: программного кода компонент, требований, системных спецификаций, функциональных спецификаций, документов проектирования и архитектуры программных систем и их компонентов, и т.д. Использование статических методов тестирования – один из наиболее эффективных способов обнаружения дефектов на ранних стадиях разработки ПО.

Действительно, статическое тестирование – это единственный способ тестирования без запуска программного кода приложения.



# Динамическое тестирование

Динамическое тестирование – процесс тестирования, производимый над работающей системой или подсистемой. Оно не может быть осуществлено без запуска программного кода приложения. Если быть более точным, динамическое тестирование состоит из:

1. запуска системы или подсистемы;
2. вызова необходимых функциональных элементов или модулей;
3. сравнения через графический интерфейс пользователя поведения системы с ожидаемым результатом поведения.



# Подходы (методы) тестирования.

Среди методов тестирования обычно выделяют несколько самых распространенных:

- метод "чёрного ящика" (Black box testing)
- метод "белого ящика" (White box or «glass-box» testing)
- метод "серого ящика" (Grey box)
- Тестирование нефункциональных аспектов программы.







# Тестирование программы как "белого ящика" и "чёрного ящика"

В терминологии профессионалов тестирования (программного и некоторого аппаратного обеспечения) фразы "тестирование белого ящика" и "тестирование черного ящика" относятся к тому, имеет ли разработчик тестов и тестирующий доступ к исходному коду тестируемого ПО, или же тестирование выполняется через пользовательский интерфейс либо прикладной программный интерфейс, предоставленный тестируемым модулем.

При тестировании белого ящика, разработчик теста имеет доступ к исходному коду и может писать код, который связан с библиотеками тестируемого ПО. Это типично для юнит-тестирования, при котором тестируются только отдельные части системы. Оно обеспечивает то, что компоненты конструкции - работоспособны и устойчивы, до определенной степени.



При тестировании чёрного ящика, тестировщик имеет доступ к ПО только через те же интерфейсы, что и заказчик или пользователь, либо через внешние интерфейсы, позволяющие другому компьютеру либо другому процессу подключиться к системе для тестирования. Например, тестирующий модуль может виртуально нажимать клавиши или кнопки мыши в тестируемой программе с помощью механизма взаимодействия процессов, с уверенностью в том, все ли идет правильно, что эти события вызывают тот же отклик, что и реальные нажатия клавиш и кнопок мыши. Как правило, тестирование чёрного ящика ведётся с использованием спецификаций или иных документов, описывающих требования к системе.



Если «альфа-» и «бета-тестирование» относятся к стадиям до выпуска продукта (а также, неявно, к объему тестирующего сообщества и ограничениям на методы тестирования), тестирование «белого ящика» и «черного ящика» имеет отношение к способам, которыми тестировщик достигает цели.

Бета-тестирование в целом ограничено техникой чёрного ящика. Таким образом, термин «бета-тестирование» может указывать на состояние программы (ближе к выпуску чем «альфа»), или может указывать на некоторую группу тестировщиков и процесс, выполняемый этой группой. Тестировщик может продолжать работу по тестированию белого ящика, хотя ПО уже на стадии «бета-тестирования»), но в этом случае он не является частью «бета-тестирования».



# Тестирование нефункциональных параметров программы.

Существуют специальные методы для тестирования аспектов программ, не являющихся функциональными, т.е. не относящихся к работоспособности самих программ.

Это тестирование:

- Тестирование производительности программного обеспечения - посмотреть работоспособность, если программа управляет большим количеством данных или имеет большое число пользователей.
- Тестирование "Юзабилити" - тестирование интерфейса пользователя, его удобства, практичности и лёгкости для освоения обычным пользователем.
- Тестирование безопасности программ важно для программ, имеющих дело с конфиденциальными данными для предотвращения использования уязвимостей хакерами.
- Тестирование качества интернационализации и локализации программного обеспечения.

Пользоваться этими методами можно и нужно, чтобы получить качественный продукт.




# Инструменты для снятия скриншотов

При проведении различных видов тестирования Вам необходимо будет фиксировать все найденные дефекты и желательно делать скриншоты или записывать дефекты на видео. Инструментов для снятия скриншотов и записи видео огромное количество. Рассмотрим некоторые из них:

При проведении различных видов тестирования Вам необходимо будет фиксировать все найденные дефекты и желательно делать скриншоты или записывать дефекты на видео. Инструментов для снятия скриншотов и записи видео огромное количество. Рассмотрим некоторые из них:

Для Windows- Print Screen (весь экран в буфере обмена) Print Screen+Alt (весь экран в буфере обмена)

Для Mac- Command+Shift+3(весь экран в файл на рабочем столе)  
Command+Shift+4(выделенная область в файл) Command+Shift+4+пробел (окно программы в файл)



Snagit- <https://www.techsmith.com/snagit.html>

Jing- <https://www.techsmith.com/jing.html>

Онлайн сервисы для обмена скриншотами:

<https://joxi.ru/>

<http://www.store-arsenal.com/>

<http://www.screencast.com/>

<http://pastenow.ru/>