ТЕСТИРОВЩИК ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

КУРС «РУЧНОЕ ТЕСТИРОВАНИЕ»

5. КЛАССИФИКАЦИЯ ВИДОВ ТЕСТИРОВАНИЯ ПО

- Уровни тестирования
- Виды тестирования ПО

Тесты существенно различаются по задачам, которые с их помощью решаются, и по используемой технике. Различие задач тестирования приводит, естественным образом, 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;
- в соответствие с традиционными показателями качества, которые проверяются с их помощью.

УРОВНИ ТЕСТИРОВАНИЯ

Модульное тестирование (Автономное или Unit-тестирование). На данном уровне тестируются по отдельности небольшие элементы системы, максимально отделенные от других элементов и, в то же время, пригодные для тестирования.

Комплексное тестирование (Сборочное тестирование, integration testing или interface testing). На данном уровне тестируются объединенные элементы (компоненты или подсистемы) общей системы, чаще всего некоторая взаимодействующая между собой группа элементов. Комплексное тестирование направлено не на проверку функционирования каждого из компонентов, а на проверку взаимодействия компонентов в соответствии с архитектурой системы.

УРОВНИ ТЕСТИРОВАНИЯ

Системное тестирование (system testing).После того, как система собрана из составляющих компонентов, она должна быть протестирована на соответствие системным спецификациям – реализованы ли все функциональные и нефункциональные требования к разрабатываемой системе. На данном уровне тестируется приложение или система (одно или более приложений) целиком.

Приемочное тестирование (Приемо-сдаточное тестирование или acceptance testing). На данном уровне завершенное приложение (система) тестируется заказчиком, конечными пользователями или соответствующими уполномоченными с целью определения соответствия системы требованиям заказчика и готовности системы к внедрению. Приемосдаточные испытания оформляют процесс передачи продукта от разработчика заказчику. В зависимости от особенностей продукта и от требований Заказчика они могут проводиться в различной форме.

УРОВНИ ТЕСТИРОВАНИЯ

Операционное тестирование (Release Testing). Даже если система удовлетворяет всем требованиям, важно убедиться в том, что она удовлетворяет нуждам пользователя и выполняет свою роль в среде своей эксплуатации, как это было определено в бизнес-модели системы. Следует учесть, что и бизнес модель может содержать ошибки. Поэтому так важно провести операционное тестирование как финальный шаг валидации. Кроме этого, тестирование в среде эксплуатации позволяет выявить и нефункциональные проблемы, такие как: конфликт с другими системами, смежными в области бизнеса или в программных и электронных окружениях; недостаточная производительность системы в среде эксплуатации и др. Очевидно, что нахождение подобных вещей на стадии внедрения – критичная и дорогостоящая проблема. Поэтому так важно проведение не только верификации, но и валидации, с самых ранних этапов разработки ПО.

УРОВНИ ТЕСТИРОВАНИЯ

Для каждого уровня тестирования могут использоваться различные виды тестирования, для каждого из которых, в свою очередь, могут использоваться различные типы тестовых испытаний.

УРОВНИ ТЕСТИРОВАНИЯ

ВИДЫ ТЕСТИРОВАНИЯ

Каждый программный продукт должен выполнять одну или несколько ключевых задач. От приложения с гео-картами мы ожидаем точной ориентации в пространстве, от сайта интернет-магазина — корректного поиска товаров по заданным параметрам и т. д. Но те же программные продукты мы можем протестировать и с точки зрения дизайна.

Таким образом, анализ ПО с позиции его ключевых или вспомогательных функций определяет тип тестирования:

- **Функциональное**
- **Нефункциональное**

ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

Функциональное тестирование направлено на проверку того, какие функции ПО реализованы, и того, насколько верно они реализованы.

НЕФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

Нефункциональное – проверка корректности работы нефункциональных требований. Оценивается, КАК программный продукт работает. Эта проверка включает в себя следующие виды:

- Тестирование производительности – работа ПО под определённой нагрузкой.
- Тестирование пользовательского интерфейса – удобство пользователя при взаимодействии с разными параметрами интерфейса (кнопки, цвета, выравнивание и т. д.).
- Тестирование UX – правильность логики использования программного продукта.
- Тестирование защищенности – определение безопасности ПО: защищено ли оно от атак хакеров, несанкционированного доступа к данным и т. д.

НЕФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

- Инсталляционное тестирование – оценка вероятности возникновения проблем при установке, удалении, а также обновлении ПО.
- Тестирование совместимости – тестирование работы программного продукта в определённом окружении.
- Тестирование надежности – работа программы при длительной средней ожидаемой нагрузке.
- Тестирование локализации –оценка правильности версии программного продукта (языковой и культурный аспекты).

СТЕПЕНЬ АВТОМАТИЗАЦИИ

В зависимости от того, используют ли тестировщики дополнительные программные средства для тестирования приложений или программ, тестирование бывает:

- **Мануальное (ручное)** – без использования дополнительных программных средств, т. е. «вручную».
- **Автоматизированное** – с использованием программных средств (более детально в описании курса по автоматизации тестирования ПО).

Каждый из подходов имеет свои преимущества и недостатки. Ручное тестирование проще освоить, оно широко применяется на проектах всех типов, но мануальные проверки отличаются монотонностью. А вот написание тестов даёт больше возможностей для творческой реализации, но автоматизация требует базовых навыков программирования.

ПОЗИТИВНОСТЬ СЦЕНАРИЯ

Этот подход определяет поведение системы в привычных и экстремальных условиях.

- **Позитивная проверка** – оценка ожидаемого поведения. Это тестирование проводится в первую очередь, ведь позволяет определить корректность работы программы.
- **Негативная** – определение устойчивости системы в нестандартной ситуации. Например, неожиданный сценарий взаимодействия пользователя с интерфейсом.

Эти типы тестирования нередко проводятся параллельно. Ведь работая над некоторой функциональностью, тестировщику проще оценить её поведение и в стандартных, и в нестандартных условиях.

ДОСТУП К КОДУ ПРОГРАММНОГО ПРОДУКТА

В процессе тестирования инженер может работать с ПО, не обращаясь к его коду, а может определить правильность работы, взглянув на код. По доступу к коду программного продукта тестирование делится на:

- **Тестирование «белого ящика»** – с доступом к коду.
- **Тестирование «черного ящика»** – без доступа к коду продукта.
- **Тестирование «серого ящика»** – на основе ограниченного знания внутренней структуры ПО. Часто говорят, что это смесь тестирования «белого ящика» и «чёрного ящика», но это в корне неверно. В данном случае тестировщик не работает с кодом программного продукта, но он знаком с внутренней структурой программы и взаимодействием между компонентами.

ИСПОЛНИТЕЛЬ

- **Альфа-тестирование** – проверка программного продукта на поздней стадии разработки. Проводится разработчиками или тестировщиками.

- **Бета-тестирование** – оценка ПО перед выходом на рынок в фокус-группе или добровольцами. Отзывы собираются, анализируются и учитываются при внесении правок.

ФОРМАЛЬНОСТЬ

Этот пункт определяет подготовленность тестировщика перед началом проверки.

- **Тестирование по тестам** – использование написанных заранее тест-кейсов.
- **Исследовательское тестирование** – одновременная разработка тестов и их использование.
- **Свободное тестирование** – проверка качества без разработки тестов и написания документации.

  Основывается на интуиции и опыте тестировщика.

Начинающие тестировщики редко работают на свободном уровне. А вот опытные QA-специалисты могут позволить себе проверку без дополнительной подготовки. Мастерство растёт со временем, как и оплата труда тестировщика

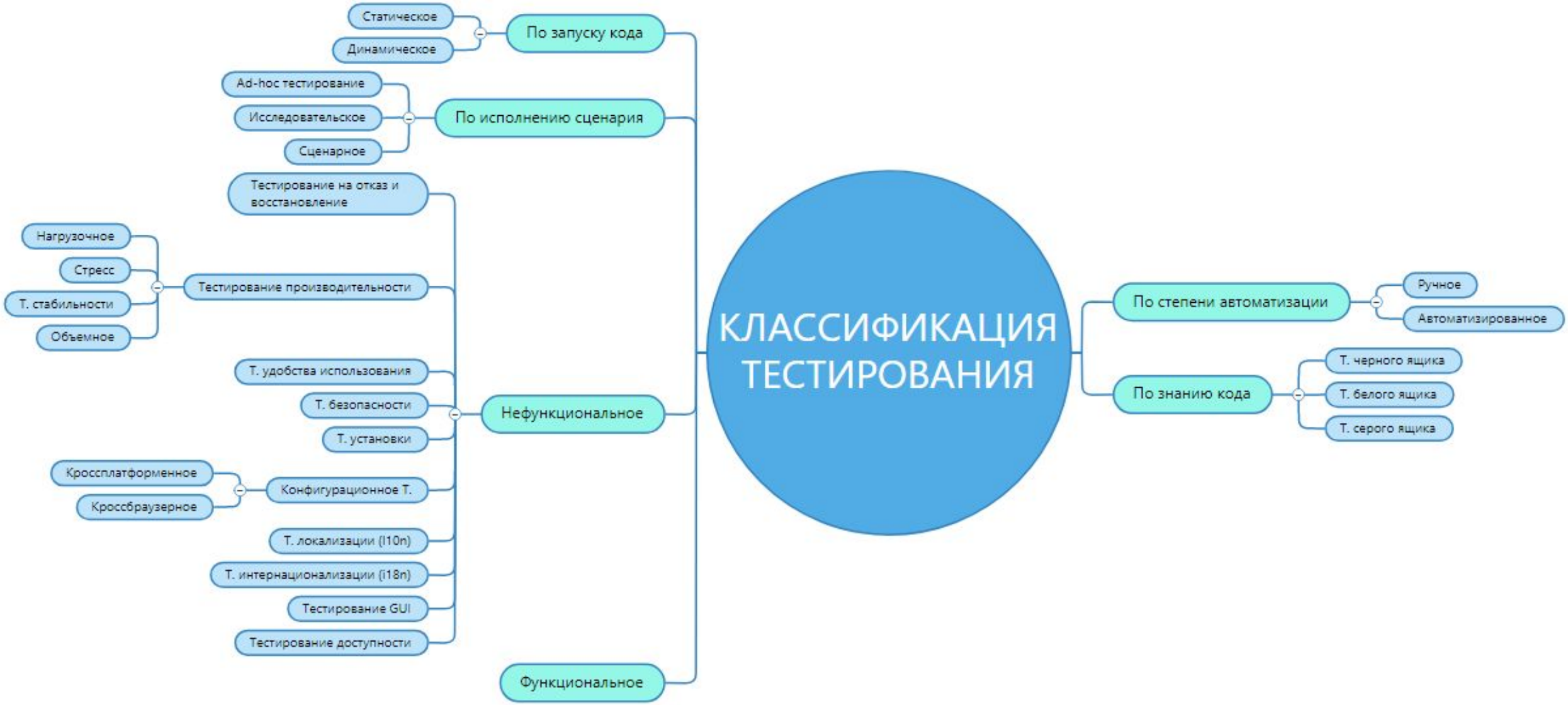
ВАЖНОСТЬ

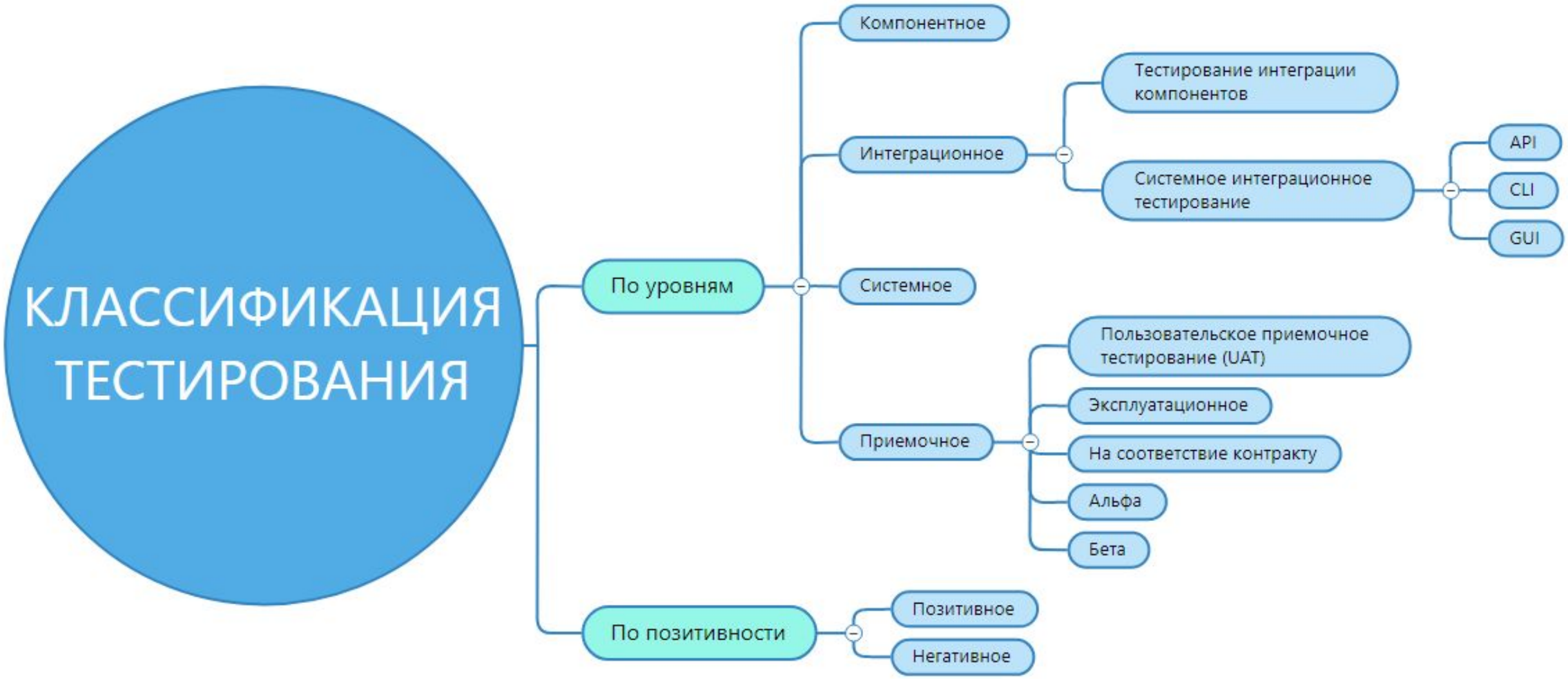
- **Дымовое тестирование** – проверка самой важной функциональности программного продукта.
- **Тестирование критического пути** – проверка функциональности, используемой типичными пользователями в повседневной деятельности.
- **Расширенное тестирование** – проверка всей заявленной функциональности.

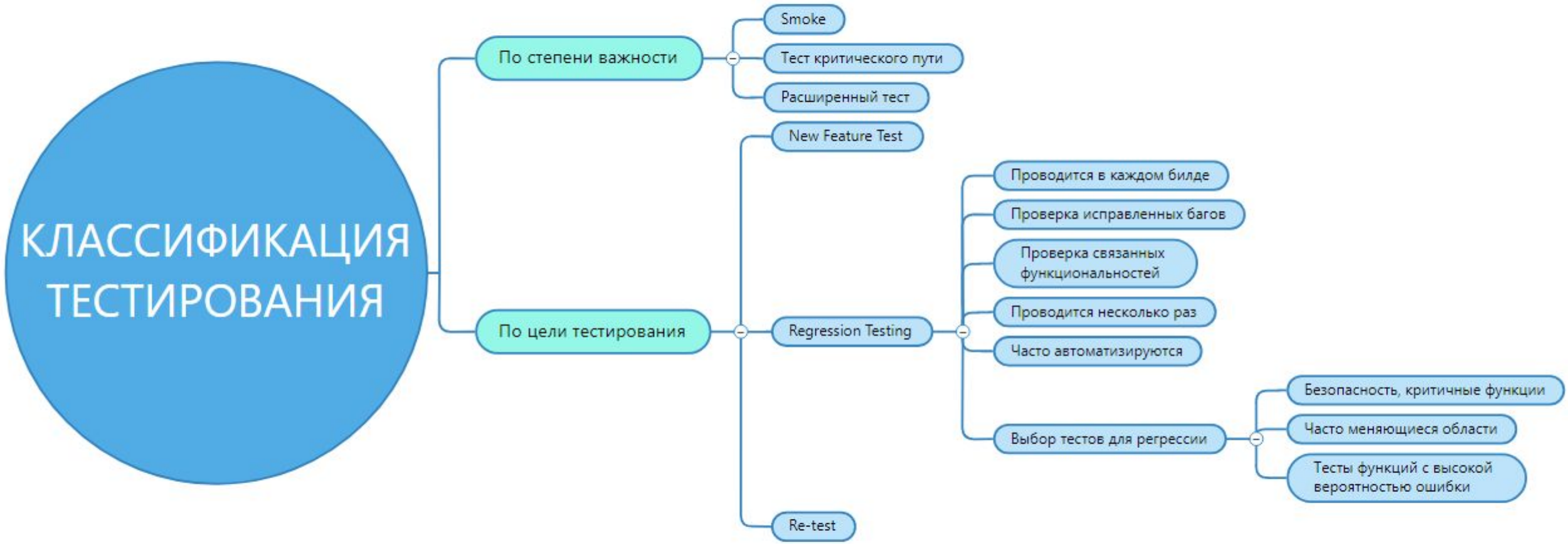
ИТОГ

QA-область очень многообразна. Помимо отличий в технологии оценки качества, тип тестирования может отличаться индустрией или видом программного продукта. К примеру, начинающий тестировщик может выбрать для себя одну из специализаций:

- тестирование мобильных или десктопных приложений;
- банкинг;
- социальные сети;
- игры
- и другое.


КЛАССИФИКАЦИЯ ТЕСТИРОВАНИЯ
По запуску кода
Статическое
Динамическое
По исполнению сценария
Ad-hoc тестирование
Исследовательское
Сценарное
Нефункциональное
Тестирование на отказ и восстановление
Тестирование производительности
Нагрузочное
Стресс
Т. стабильности
Объемное
Т. удобства использования
Т. безопасности
Т. установки
Конфигурационное Т.
Кроссплатформенное
Кроссбраузерное
Т. локализации (l10n)
Т. интернационализации (i18n)
Тестирование GUI
Тестирование доступности
Функциональное
По степени автоматизации
Ручное
Автоматизированное
По знанию кода
Т. черного ящика
Т. белого ящика
Т. серого ящика


КЛАССИФИКАЦИЯ ТЕСТИРОВАНИЯ
По уровням
Компонентное
Интеграционное
Тестирование интеграции компонентов
Системное интеграционное тестирование
API
CLI
GUI
Системное
Приемочное
Пользовательское приемочное тестирование (UAT)
Эксплуатационное
На соответствие контракту
Альфа
Бета
По позитивности
Позитивное
Негативное


КЛАССИФИКАЦИЯ ТЕСТИРОВАНИЯ
По степени важности
Smoke
Тест критического пути
Расширенный тест
По цели тестирования
New Feature Test
Regression Testing
Проводится в каждом билде
Проверка исправленных багов
Проверка связанных функциональностей
Проводится несколько раз
Часто автоматизируются
Выбор тестов для регрессии
Безопасность, критичные функции
Часто меняющиеся области
Тесты функций с высокой вероятностью ошибки
Re-test

ТЕСТИРОВЩИК ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

КУРС «РУЧНОЕ ТЕСТИРОВАНИЕ»

5. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ

Изучение методов, видов и типов тестирования на примере «Тестирования карандаша»

ВОПРОС НА СОБЕСЕДОВАНИИ: ПРОТЕСТИРУЙТЕ ОБЫЧНЫЙ КАРАНДАШ
ВАРИАНТ ПОДГОТОВКИ ОТВЕТА:

I. Есть требования или спецификация?

II. У заказчика есть специфические потребности (привычка грызть карандаш, специальные условия работы)?

III. Краткий план тестирования: обсуждаем

Объект тестирования – простой карандаш со стирательной резинкой на конце (или без), специальных требований нет.

Рамки тестирования – что мы считаем отказом. Это нужно для стресс-тестов.

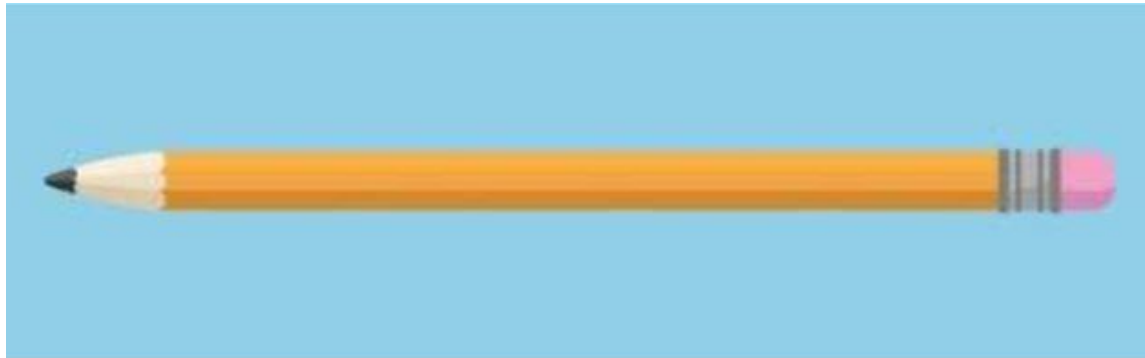
Критерии верификации – мы анализируем результаты тестирования? Если да – какие критерии верификации?

Сроки – разумный срок на собеседовании – 5-10 минут.

Ресурсы – 1 ручной тестер, простой карандаш как черный ящик, бумага, точилка (было бы полезно).

IV. Составляем примерные требования.

1. Простой карандаш пишет на бумаге.
2. Наш карандаш имеет на конце стирательную резинку.
3. Резинка позволяет стереть написанное.
4. Можно заточить точилкой.
5. Бывает мягким и твердым. У нас _____



ВОПРОС НА СОБЕСЕДОВАНИИ: ПРОТЕСТИРУЙТЕ ОБЫЧНЫЙ КАРАНДАШ
ВАРИАНТ ПОДГОТОВКИ ОТВЕТА:

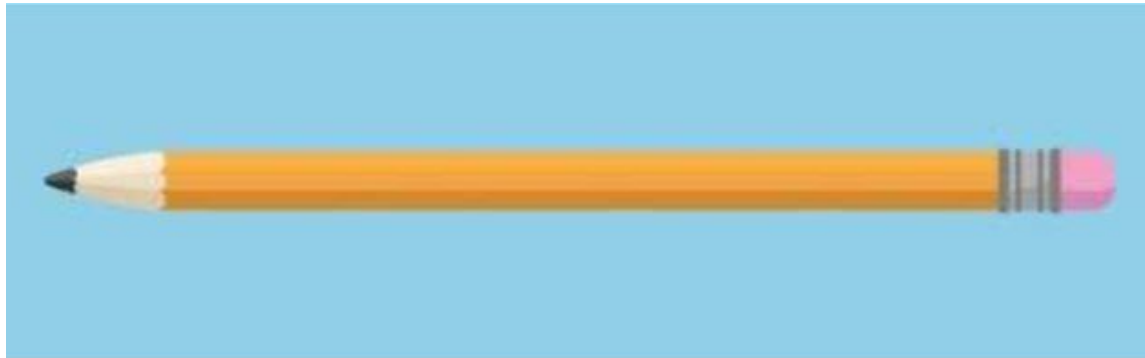
V. Список видов тестирования и тест-кейсов:

1. **Функциональное** тестирование:
   - Позитивное!!! Смотрим требования!
   - Негативное. Его делаем в конце!
2. **Надежность:** как часто ломается грифель. А есть ли время? Ожидаемое поведение?
3. **Юзабилити:** удобно держать в руке, мягко пишет, красивый, не скатывается со стола.
4. **Производительность:** сколько испишем при среднем нажиме, как быстро придется точить. А есть ли время? Ожидаемое поведение?
5. **Конфигурационное:** подходит для правой и левой руки, подходит для разного типа бумаги, подходит для разных точилок (если они есть).
6. **Стрессовое:** при сильном нажиме.

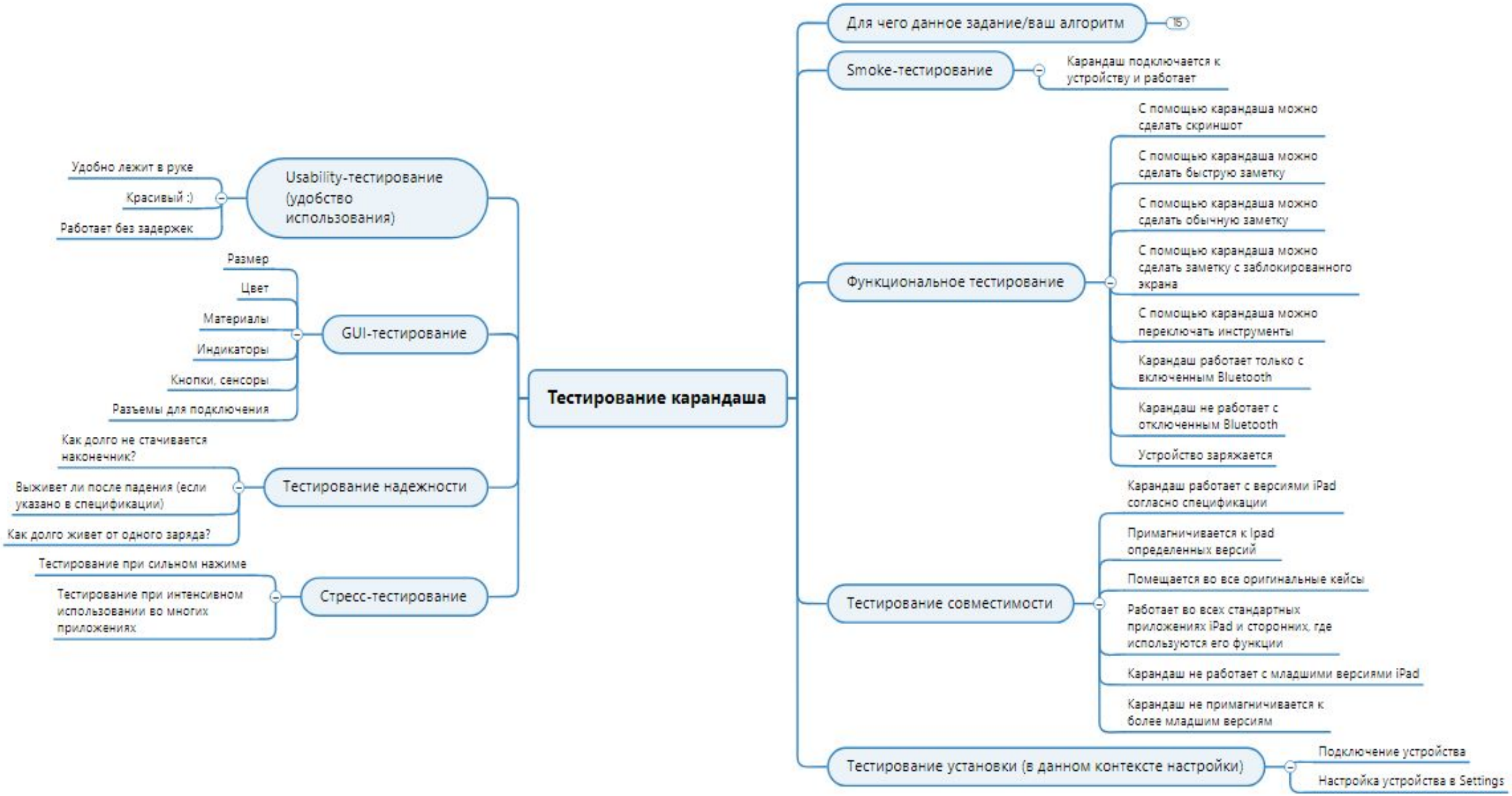
VI. Делаем выводы.

Какие ошибки мы нашли в процессе тестирования?

Обладает ли карандаш приемлемым для использования качеством?



А ТЕПЕРЬ РАССМОТРИМ ТЕСТИРОВАНИЕ НЕ ПРОСТОГО КАРАНДАША, А APPLE PENCIL. СНАЧАЛА ПОПРОБУЙТЕ СДЕЛАТЬ ЭТО САМИ, НЕ ГЛЯДЯ НА ОТВЕТ


Тестирование карандаша
Для чего данное задание/ваш алгоритм
15
Smoke-тестирование
Карандаш подключается к устройству и работает
Функциональное тестирование
С помощью карандаша можно сделать скриншот
С помощью карандаша можно сделать быструю заметку
С помощью карандаша можно сделать обычную заметку
С помощью карандаша можно сделать заметку с заблокированного экрана
С помощью карандаша можно переключать инструменты
Карандаш работает только с включенным Bluetooth
Карандаш не работает с отключенным Bluetooth
Устройство заряжается
Тестирование совместимости
Карандаш работает с версиями iPad согласно спецификации
Примагничивается к Ipad определенных версий
Помещается во все оригинальные кейсы
Работает во всех стандартных приложениях iPad и сторонних, где используются его функции
Карандаш не работает с младшими версиями iPad
Карандаш не примагничивается к более младшим версиям
Тестирование установки (в данном контексте настройки)
Подключение устройства
Настройка устройства в Settings
Usability-тестирование (удобство использования)
Удобно лежит в руке
Красивый :)
Работает без задержек
GUI-тестирование
Размер
Цвет
Материалы
Индикаторы
Кнопки, сенсоры
Разъемы для подключения
Тестирование надежности
Как долго не стачивается наконечник?
Выживет ли после падения (если указано в спецификации)
Как долго живет от одного заряда?
Стресс-тестирование
Тестирование при сильном нажиме
Тестирование при интенсивном использовании во многих приложениях