Это еще не конец!

Чеклист Майкла Хантера

Перевод: Ольга Алифанова

Оригинал: http://www.thebraidytester.com/downloads/YouAreNotDoneYet.pdf

Это еще не конец!

Выберите что-нибудь, что угодно – функцию в вашем любимом приложении, игрушку, предмет мебели. Подумайте о том, что можно сделать, чтобы протестировать это, придумайте максимальное количество тестов, которые только придут вам в голову. Продолжите чтение, когда закончите.

Что, вы снова здесь? Я гарантирую, что есть тесты, о которых вы не подумали. Откуда я знаю? Оттуда, что даже для крошечной части продукта – например, для диалогового окна поиска в вашем браузере – можно придумать миллиарды тестов. Некоторые из них найдут интересные баги с большей вероятностью, некоторые с меньшей. Некоторые из них стоит провести, чтобы убедиться, что функциональность работает корректно. Такие кейсы – основа моего списка "Это еще не конец".

Это довольно большой, пугающе выглядящий список. Без паники! Возможно, многие тесты вы уже провели, а какие-то просто неприменимы к вашей конкретной ситуации. Возможно, ряд из них вы не проводите сознательно, и у вас есть для этого основания. Цель этого списка – спровоцировать размышления о том, какие тесты вы уже провели, а какие нет, и выявить области, которые вы собирались покрыть в своем тестировании, но пока не покрыли.

Не вздрагивайте при мыслях о тестах, которые вы еще не проводили. Подгоните мой список под свой контекст. Выбросите все, что неприменимо, используйте его, как стартовую точку для поиска моментов, которые применимы к вашей ситуации, но не вошли в мой список. Пусть он станет основой организации вашего тестирования или чек-листом, по которому вы пробегаетесь под конец тестирования. Неважно, как вы его используете – просто пользуйтесь им.

Методы ввода

Вы не закончили тестировать, если не проверили эти методы ввода:

- Клавиатура. Пфф, наверное, думаете вы. Но проверка ввода с клавиатуры не ограничивается проверкой возможности ввода символов в текстовые поля. Найдите абсолютно все в вашем приложении, способное принимать текстовый ввод. Это необязательно значения – это еще и сочетания клавиш, и навигация (да, этот раздел немного перекликается с разделом "Навигация по диалоговым окнам и доступность"). Если ваше приложение использует специфичные функции управления, обратите на них особое внимание – скорее всего, они особенным образом воспринимают ввод с клавиатуры. Пусть ненавидящие мышь мастера клавиатуры будут счастливы!
- Мышь. Еще одно "Пфф", но проблема в том, что это настолько очевидно, что легко упускается из виду. Повторюсь, обращаем особое внимание на специфические функции – возможно, они иначе реагируют на манипуляции при помощи мыши.
- Перо. Это может означать прямой ввод данных в приложение, фильтрацию через операционную систему (например, через панель ввода с планшета в Microsoft Windows), или фильтрацию через панели ввода стороннего ПО – зависит от вашей платформы. У каждого метода ввода есть свои особенности, которые могут конфликтовать с особенностями вашего приложения.
- Голосовой ввод. В зависимости от платформ, для которых предназначено ваше приложение, это может быть прямой ввод, фильтрация через ОС, или через сторонние приложения.
- Ввод на иностранных языках. В Windows это обычно связано с использованием редактора методов ввода (IME) – или того, который поставляется вместе с ОС, или стороннего. Это может вызывать затруднения даже в приложениях, не использующих специфичную обработку данных с клавиатуры. К примеру, процессор ввода символов на японском языке, скорее всего, захватывает нажатия, комбинирует несколько нажатий в один иероглиф, и затем отправляет его в приложение. Сочетания клавиш должны игнорироваться при такой обработке, но зачастую этого не происходит. Да, можно просто выключить IME, но почти всегда это плохое решение.
- Вспомогательные методы ввода (например, дыхательные трубки). Обычно операционная система воспринимает их, как стандартную клавиатуру или мышь, но они могут спровоцировать специфические ситуации, которые должны обрабатываться приложением – например, очень длительная задержка между нажатиями клавиш.
- Другие методы ввода. К примеру, я сталкивался с играми, управляющимися через сенсоры, надетые на пальцы. Иногда такие устройства определяются как джойстик или мышь. Что произойдет, если ваше приложение будет управляться при помощи такой штуки?
- Несколько клавиатур и/или мышей. Windows поддерживает несколько подобных устройств одновременно. Единовременно у вас будет только один курсор, поэтому вам не придется гадать, как поддерживать одновременный ввод с разных клавиатур. Однако стоит поразмыслить над возможными "прыжками" с одного устройства на другое. Мечта тестировщика!

Файлы

Вы не закончили тестировать, если вы не обратили внимания на все возможные файлы в вашем приложении – иногда они переполнены информацией, которая просто игнорируется. Все мы знаем, что бывает, когда что-то игнорируется. Баги! Помню баг, на который разработчик натолкнулся раз пятьдесят, просто пройдясь по этому чек-листу.

- Убедитесь, что версия каждого файла верна.
- Убедитесь, что версия сборки верна для каждой возможной сборки. Обычно версия сборки и версии файлов совпадают, но определяются при помощи различных механизмов и должны быть синхронизированы.
- Убедитесь, что информация об авторских правах на каждый файл верна.
- Убедитесь, что файлы имеют цифровую подпись (или не имеют, если не должны). Убедитесь, что подпись верна.
- Убедитесь, что файлы находятся в нужных папках (см. раздел "Установка").
- Убедитесь, что знаете, от чего зависит каждый файл. Проверьте, что все зависимости или обеспечены при установке, или гарантированно присутствуют на компьютере.
- Проверьте, что происходит, если какой-то из файлов – или какая-то зависимость между файлами – отсутствует.
- Проверьте текст в каждом файле и убедитесь, что с тем, что этот текст увидит пользователь, нет никаких проблем.

Названия файлов

Вы не закончили тестировать, если не проверили варианты названий файлов:

- Из одного символа.
- Короткие.
- Длинные.
- Очень длинные.
- Названия с использованием проверок для текстовой строки.
- Названия, содержащие зарезервированные слова.
- Название "file" (file.ext)
- Полный путь к файлу (c:\My\Directory\Structure\file.ext)
- Относительный путь к подпапке (Sub\Folder\file.ext)
- Относительный путь к папке (.\file.ext)
- Относительный путь к родительской папке (..\Parent\File.ext)
- Путь с множественными вложениями (Some\Very\Very\Very\Very\Very\Deeply\Nested\File\That\You\Will\Never\Find\Again\file.ext).
- Сетевой путь UNC (\\server\share\Parent\file.ext).
- Путь к диску (Z:\Parent\file.ext).

Названия файлов – частый источник багов. Приложения Microsoft Windows, не защищенные от использования зарезервированных слов, напрашиваются на Denial of Service. Приложения, позволяющие открытие, сохранение и изменение любых файлов, оставляют дыру для манипуляций с "защищенными" файлами. Некоторые пользователи пихают все созданные

документы в пользовательскую папку. Другие создают отдельную папку для каждого. В названии файлов разрешены символы, которые более не разрешены нигде, и наоборот. Покопайтесь в этой области – возможно, ваши труды окупятся.

Запрещенные символы в названиях файлов, другие ошибки

Вы не закончили тестировать, если не проверили, как приложение работает с запрещенными символами в названиях файлов и зарезервированными названиями. Операционные системы не любят, когда вы используете "джокеров" – например, символ * - в названиях файлов, и могут специфически воспринимать ряд других названий. Скажем, Windows предоставляет единый API для создания и открытия файлов, коммуникационных портов, и других кросспроцессных механизмов коммуникации. Распространенные коммуникационные порты – к примеру, COM1 – воспринимаются как название файла, как будто это на самом деле файл. Это удобно, но это значит, что вы не можете назвать физический, присутствующий на диске файл COM1.

Это довольно легко тестируется – набросайте список интересных идей, и вставьте каждый вариант во все возможные диалоговые окна, командные строки и API, принимающие названия файлов. Запрещенные символы, скорее всего, вызовут сообщение об ошибке, но попытка открыть файл с зарезервированным именем может подвесить приложение.

Тема на MSDN про названия файлов: https://msdn.microsoft.com/ru-ru/library/windows/desktop/aa365247

Операции с файлами

Вы не закончили тестировать, если не проверили со всей возможной тщательностью функциональность открытия, сохранения, и "сохранения как". Не знаю, как у вас, но лично меня бесит, когда плоды моих трудов вылетают в трубу! Для ряда приложений отсутствие возможности сохранения и последующего открытия данных означает, что эти приложения не имеют никакого смысла. Следовательно, очень важно проверить, что все работает нормально при следующих условиях:

- Откройте каждый поддерживаемый тип/версию файлов, и сохраните через "Сохранить как" в каждом возможном типе/версии. Очень важно проверить, что открытие и сохранение файлов, созданных в предыдущих версиях, работает. Пользователи бесятся, когда переход на новую версию означает, что старые документы нельзя открыть. Они не будут обновлять приложение, если не смогут легко делиться документами с теми несчастными, которые все еще пользуются старыми версиями.
- Откройте все поддерживаемые типы файлов и нажмите "Сохранить". Если в процессе сохранения (а не через "Сохранить как") можно выбрать версию и тип файла, сделайте то же, что и в предыдущем сценарии. Как правило, меню "Сохранить" дает сохранить только актуальную версию.
- Превратите файл предыдущей версии в современную, а потом сохраните его вновь как предыдущую версию. Откройте результат. Он верно открывается? Новые возможности приложения правильно сконвертированы в то, что понимает и предыдущая версия? Как обрабатываются встроенные объекты предыдущих версий?
- Откройте файл, созданный в более новой версии, в предыдущей версии приложения. Если документ открывается, как выглядит то, что создано при помощи новых возможностей?

Если он не открывается, есть ли там внятное сообщение об ошибке, указывающее, почему?

- Проверьте функции "Открыть", "Сохранить", "Сохранить как" в разных файловых системах (FAT, NTFS) и протоколах (локальный диск, UNC, http://). Операционная система обычно не отображает разницу между типами файловых систем, но за разные протоколы могут отвечать разные участки кода вашего приложения.
- Проверьте работу "Открыть", "Сохранить" и "Сохранить как" следующим образом (если это применимо):
    - Через меню
    - Через панель инструментов
    - Через сочетание клавиш (например, CTRL+S для сохранения)
    - Через список "Недавно открытые"
    - Через библиотеку документов Microsoft SharePoint
    - Через контекстное меню
    - Через список "Недавно открытые" внутри приложения.
    - Через список "Недавно открытые" в операционной системе.
    - Перетащите из проводника.
    - Перетащите с рабочего стола.
    - Перетащите из другого приложения.
    - Через командную строку.
    - Через двойной клик на ярлыке рабочего стола.
    - Через двойной клик на ярлыке в письме или другом документе.
    - Через встроенные объекты.
- Проверьте работу "Открыть", "Сохранить" и "Сохранить как" для:
    - Файлов, изменения которых разрешены.
    - Файлов, для которых допускается только чтение.
    - Файлов, к которым у вас нет доступа (например, политики безопасности настроены так, что вы не можете получить доступ к этому файлу).
    - Папок, в которые можно сохранять изменения.
    - Папок, доступных только для чтения.
    - Жесткого диска
    - Переносного диска
    - USB-диска
    - CD-ROM
    - CD-RW
    - DVD-ROM
    - DVD-RW.
- Проверьте работу "Открыть", "Сохранить" и "Сохранить как" для разных типов и скоростей сетевых соединений. Модемы и широкополосной доступ различаются по характеристикам и необязательно равны вашим офисным, сто-гигабайт-в-секунду скоростям!
- Откройте файлы, созданные в (и попытайтесь их там сохранить):
    - Других операционных системах
    - ОС с другим системным языком
    - ОС с другим пользовательским языком
    - Версии вашего приложения на другом языке.
- Откройте/сохраните/"сохраните как" файлы с названиями

    - Из списка раздела "Текстовые поля"
    - Из списка раздела "Названия файлов"
    - Из списка раздела "Недопустимые названия файлов"
    - С пробелами.
- Сделайте следующее в процессе открытия/сохранения:
    - Оборвите сетевое соединение.
    - Переключитесь на другое сетевое соединение.
    - Перезагрузите приложение
    - Перезагрузите компьютер.
    - Отправьте машину в спящий режим.
    - Отправьте машину в режим гибернации.
- Проверьте автосохранение. Что произойдет, если оно настроено на "сохранять каждые 0 минут"? А если на каждую минуту? А если документ очень большой? Если автосохранение настраивается отдельно для каждого документа, что произойдет, если несколько автосохранений запустятся одновременно, или одно запустится, когда другое уже стартовало? Работает ли функция восстановления файлов так, как ожидается? Что произойдет, если приложение упадет в ходе автосохранения? А если оно упадет при попытке восстановить автосохраненный документ?
- Сохраните/"Сохраните как" при следующих условиях:
    - В файлах нет изменений.
    - В одном файле есть изменения.
    - В нескольких файлах есть изменения, и пользователь хочет сохранить все.
    - В нескольких файлах есть изменения, и пользователь не хочет сохранять ни один из них.
    - В нескольких файлах есть изменения, и пользователь хочет сохранить только определенные файлы.

Сетевое соединение

Вы не закончили тестировать, если вы не проверили, как ваше приложение работает с различными сетевыми конфигурациями и событиями. Раньше можно было рассчитывать на относительную стабильность сети - если компьютер в сети, когда приложение стартовало – скорее всего, он останется подключенным, пока оно работает. Конечно, какой-нибудь придурок мог случайно задеть кабель ногой или выдернуть не тот шнур из роутера, но риск какой-нибудь катастрофы в этой области был довольно низким – достаточно низким, чтобы баги в духе "Отключите ваш компьютер от сети, когда приложение открывает файл в 20 мегабайт из сетевой папки" откладывались как Won't Fix со словами "Пользователь так не делает".

О, как изменился мир! Современные пользователи зачастую пользуются беспроводными сетями, которые могут обвалиться в любой момент. Пользователи, начавшие работу в проводной сети, могут отключить провод от ноутбука и оборвать таким образом соединение. К тому же они могут пользоваться сетевым соединением через мобильный телефон. Бывшие "будет время – поглядим"-проблемы, связанные с сетью, стали реальностью, регулярно раздражающей ваших пользователей. Поэтому проверьте вот что:

- Сетевое соединение, поддерживающее только IPv4
- Сетевое соединение, поддерживающее только IPv6

- Сетевое соединение, поддерживающее IPv4 и IPv6
- Соединение через беспроводную сеть 802.11a
- Соединение через беспроводную сеть 802.11b
- Соединение через беспроводную сеть 802.11g
- Соединение через беспроводную сеть 802.11n
- Соединение через GPRS
- Соединение через машину, подключенную к нескольким сетям.
- Соединение через модем 28.8
- Соединение через модем 56k
- Соединение через сеть, находящуюся вовне вашего корпоративного файервола
- Соединение через сеть, которая требует аутентификации при первом подключении.
- Соединение через сеть, которая требует аутентификации каждый раз.
- Доступ через файервол на стороне ПО.
- Доступ через файервол на стороне железа.
- Доступ через Network Address Translation.
- Потеря соединения с сетью.
- Потеря прав на соединение с сетью.
- Присоединение к рабочей группе.
- Присоединение к домену.
- Доступ к документам в сетевом окружении, требующий аутентификации.
- Предварительный просмотр документа для сетевого принтера, который в данный момент недоступен.

Сообщения об ошибках

Вы не закончили тестировать, если вы не проверили все сообщения об ошибках, уведомления и предупреждения и диалоговые окна в вашем приложении, и не убедились в следующем:

Содержание

- Убедитесь, что вы понимаете, чем вызвано появление предупреждения, и что ваши тест-кейсы покрывают все условия его появления (или вы намеренно решили не проверять какие-то из этих условий).
- Убедитесь, что предупреждение на самом деле необходимо. К примеру, если действие обратимо, дополнительного подтверждения уверенности пользователя в том, что он хочет его совершить, не требуется.
- Убедитесь, что предупреждение сначала сообщает о проблеме, а потом информирует о решении. Обращайтесь с пользователями, как с умными, знающими людьми, помогите им понять, в чем проблема и как они могут ее решить.
- Убедитесь, что тон предупреждения вежлив и доброжелателен, а не обвиняет пользователя. Помню, как-то раз приложение, восстанавливающее свою базу данных после краша, упрекнуло меня в том, что я его неправильно закрыл. Эм. Нет, это ТЫ неправильно закрылось! Я не имею к этому никакого отношения! В более поздней версии текст уведомления изменился на "Это приложение было неправильно закрыто", и стало чуть лучше. Скорее всего, ваш пользователь не пытался создать проблемную ситуацию намеренно. Если он сделал то, что сделал, намеренно – скорее всего, он не знал, что это вызовет проблему. Сообщите ему, что произошло, что предпринимает приложение, чтобы

исправить ситуацию, и как избежать такой ситуации впредь.

- Убедитесь, что текст сообщения верен и подходит для конкретной ситуации.
- Убедитесь, что текст сообщения выдержан в едином стиле и соответствует другим сообщениям системы.
- Убедитесь, что сообщение достаточно подробно, и при этом не вдается в излишние детали. Хозяйке на заметку: если оно занимает более трех строк текста, то, скорее всего, оно чересчур длинное.
- Убедитесь, что сообщение содержит полные, законченные предложения, и в нем не упущены заглавные буквы и знаки препинания.
- Убедитесь, что оно не содержит аббревиатур и сокращений. Специфические аббревиатуры допускаются, только если вы абсолютно убеждены, что все ваши пользователи поймут, о чем речь.
- Убедитесь, что сообщение использует название продукта, а не местоимения вроде "я" или "мы".

Функциональность:

- Убедитесь, что заголовок сообщения содержит название продукта (например, "Acme Word Processor").
- Убедитесь, что все кнопки корректно работают. Я сталкивался с кучей кнопок "Отмены", которые на самом деле были замаскированными кнопками "ОК"!
- Убедитесь, что для каждой кнопки есть уникальное сочетание клавиш.
- Убедитесь, что кнопки находятся после текста, а не до него.
- Убедитесь, что графические элементы сообщения правильно размещены и подходят к ситуации. Для приложений Microsoft Windows существуют стандартные иконки информационных, предупреждающих и критических уведомлений, и обычно они расположены слева от текста сообщения.

Доступность

Вы не закончили тестировать, если не убедились, что ваше приложение умеет работать с функциями доступности, встроенными в вашу операционную систему. Эти функции важны для слепых и глухих, а также тех, кто использует вспомогательные устройства ввода, но ими также пользуются и другие люди. К примеру, поддержка крупных шрифтов будет с благодарностью встречена пользователями со слабым зрением, и/или работающим на мониторах с высоким DPI.

Некоторые элементы этого списка специфичны для Microsoft Windows, но в других операционных системах наверняка есть нечто похожее.

- Убедитесь, что каждый элемент управления в каждом диалоговом окне и любой другой части пользовательского интерфейса поддерживает как минимум эти свойства Microsoft Active Accessibility (MSAA):
    - Название – его идентификатор.
    - Роль – описание того, что элемент делает (активен ли он, принимает ли он значения).
    - Состояние – описание актуального состояния элемента.
    - Значение – текстовое представление актуального состояния.
    - Сочетание клавиш – комбинация, которую можно использовать для фокуса на этом

элементе.
    - Действие по умолчанию – описание того, что произойдет, если пользователь вызовет этот элемент. К примеру, для заполненного чекбокса действие по умолчанию – очистить, а для кнопки – нажать.
- Убедитесь, что изменение значения каждого элемента меняет значения "Состояние" и "Значение" в MSAA.
- Запустите приложение в режиме высокой контрастности, в котором доступно всего несколько цветов вместо полной палитры. Приложение работает, им можно пользоваться? Смена статусов и другие UI-элементы видимы глазу? Панели инструментов и другие элементы можно разглядеть и прочитать? Есть ли в приложении что-то, не поддерживающее такой режим?
- Запустите приложение в режиме увеличенных шрифтов, когда системные шрифты установлены крупными. Убедитесь, что меню, диалоговые окна и другие элементы поддерживают этот режим, и их можно прочитать. Обратите особое внимание на текст, обрезанный по горизонтали или вертикали! Чтобы по-настоящему испытать свой интерфейс, проверьте это на псевдолокализованном билде.
- Запустите приложение с Sound Sentry, которое отображает информационное сообщение, мигает, или другим способом предупреждает пользователя, что в приложении проигрывается звук. Убедитесь, что любые звуки вашего приложения активируют Sound Sentry.
- Запустите приложения в режиме залипания клавиш, позволяющего пользователю вводить сочетания клавиш по одной, а не одновременно. Операционная система скроет подробности пользовательского ввода от приложения, но если оно напрямую отслеживает состояние нажатых клавиш, то, возможно, такую ситуацию нужно обрабатывать отдельно.
- Запустите приложение, используя режимы, позволяющие манипулировать курсором мыши и кнопками при помощи цифровой клавиатуры. Операционная система скроет подробности пользовательского ввода от приложения, но если оно напрямую отслеживает состояние мыши, то, возможно, такую ситуацию нужно обрабатывать отдельно.
- Запустите приложение, отключив мышь, и убедитесь, что абсолютно все элементы интерфейса доступны с клавиатуры. Любой тест, который вы проводите при помощи мыши, должен сработать и в этом режиме.
- Запустите приложение в режиме чтения с экрана, и выключите монитор. Вы должны быть способны выполнить любой из ваших кейсов в таком режиме.
- Убедитесь, что приложение сообщает о приобретении и потере фокуса элементами.
- Убедитесь, что переход по диалогам и другим доступным по TAB элементам соответствует их реальному расположению.
- Убедитесь, что любым цветовым индикаторам (как, например, волнистой линии, которой Word сигнализирует об орфографических ошибках) можно поменять цвет.
- Убедитесь, что мерцание у мерцающих объектов соответствует мерцанию курсора, установленному в системе.

Полнота поддержки этой функциональности, конечно, зависит от бизнес-решений, принятых вашей командой. Приложения, связанные с рисованием и работой с другой графикой, обычно требуют мыши для собственно рисования. Однако ряд проверок, специфичных для доступности, можно применять и в других сценариях. К примеру, многие пользователи используют клавиатуру

для передвижения элементов по экрану в программах для рисования.

Доступность текста

Вы не закончили тестировать, если вы не убедились, что весь текст в вашем приложении – это текст, а не картинка и не видео. Текст в форме картинки может вызвать два типа проблем. Во-первых, люди, использующие чтение экрана, не узнают, что написано на ваших изображениях, видео и анимациях. Во-вторых, текст, встроенный в графику, усложняет локализацию продукта. Перевод требует простой модификации исходных файлов приложения, а вот перевод изображений и видео требует их пересоздания.

Если вы никак не можете этого избежать, облегчите жизнь локализаторам, создавая такие изображения динамически с использованием текстовых строк. Видео и анимация тоже могут создаваться таким образом, все зависит от вашего инструментария.

Что касается проблем доступности, убедитесь, что соответствующая информация доступна каким-либо иным образом – через вспомогательный текст, через alt-тэги HTML, и так далее.

Меню и панели инструментов

Вы не закончили тестировать, если вы не проверили работу ваших меню и панелей инструментов. Раньше они явным образом различались: меню могли содержать подменю и всегда были текстовыми (возможно, с опциональными иконками), а панели инструментов никогда не содержали подменю и были исключительно графическими. Сейчас это практически одно и то же, и единственное реальное различие между ними в том, что панель инструментов видна всегда, а меню – нет.

- Убедитесь, что все команды меню и панелей инструментов работают.
- Убедитесь, что все сочетания клавиш работают.
- Убедитесь, что встроенные команды работают из пользовательских меню.
- Убедитесь, что встроенные команды работают из пользовательской панели инструментов.
- Убедитесь, что пользовательские команды работают из пользовательского меню.
- Убедитесь, что пользовательские команды работают из пользовательской панели инструментов.
- Убедитесь, что пользовательские команды работают из встроенного меню.
- Убедитесь, что пользовательские команды работают из встроенной панели инструментов.
- Убедитесь, что пользовательские меню и панели правильно сохраняются и загружаются.
- Убедитесь, что изменения, внесенные во встроенные меню и панели инструментов, правильно применяются и отображаются.
- Убедитесь, что команды скрываются или становятся неактивными, и показываются/становятся доступными тогда и только тогда, когда это допустимо.
- Убедитесь, что описания команд верны и соответствуют терминологии, использующейся в других местах.
- Убедитесь, что контекстные меню элементов меню и панелей работают верно.
- Убедитесь, что текст в строке состояния верен.
- Убедитесь, что текст в строке состояния не обрезается.

Поведение диалоговых окон

Вы не закончили тестировать, если вы не проверили следующие моменты, работая с диалоговыми окнами:

- Убедитесь, что каждая команда (элемент меню, сочетание клавиш, и т. д.), которая инициирует диалоговое окно, открывает его.
- Убедитесь, что заголовок окна верен.
- Убедитесь, что терминология, использованная в тексте диалогового окна, соответствует терминологии, использующейся в приложении.
- Убедитесь, что принятие диалогового окна приводит к правильным изменениям состояния приложения.
- Убедитесь, что отмена диалогового окна не меняет состояния приложения.
- Убедитесь, что диалоговое окно запоминает свою позицию и открывается в том месте, где оно закрывалось в последний раз. Или, как вариант, что оно всегда отображается на одном и том же месте, если оно не должно запоминать свою позицию.
- Убедитесь, что содержание диалогового окна или отображает состояние приложения, или всегда имеет значения по умолчанию, если это окно не зависит от текущего состояния приложения.
- Убедитесь, что вызов помощи (например, нажатие F1) открывает соответствующий раздел помощи. Обратите внимание, что это, возможно, нужно проверить для каждого элемента, так как у некоторых диалоговых окон есть специфическая контекстная помощь для элементов управления.

Интерактивность диалоговых окон

Вы не закончили тестировать, если вы не проверили диалоговые окна как следует:

- Убедитесь, что в заголовке диалогового окна отображаются нужные контрольные элементы (например, некоторые окна можно увеличивать, а некоторые – нет), и что они работают как надо.
- Убедитесь, что дефолтные положения элементов, отвечающих за фокус и редактирование, находятся в правильном положении.
- Убедитесь, что диалоговое окно можно отменить, нажав:
    - Escape (вне зависимости от положения фокуса на контрольных элементах)
    - Системную кнопку "Закрыть" (крестик в диалоговых окнах MS Windows)
    - Кнопку отмены на диалоговом окне.
- Убедитесь, что навигация через TAB следует в правильном порядке.
- Убедитесь, что у всех контрольных элементов есть сочетание клавиш, позволяющее их активировать, что оно работает, и что оно уникально в рамках этого диалогового окна.
- Убедитесь, что подсказки для всех элементов окна верны.
- Убедитесь, что взаимоисключающие контрольные элементы правильно работают одновременно.
- Проверьте все состояния диалогового окна (например, различные наборы контрольных элементов, отображающиеся в зависимости от состояния системы, или кнопки "Развернуть" и "Свернуть").
- Убедитесь, что все контрольные элементы, которые могут принимать неопределенное

состояние (например, кнопка полужирного начертания примет неопределенное состояние, если в выбранном тексте есть как полужирное начертание, так и какое-либо другое), работают корректно.

- Убедитесь, что редактирование неопределенного значения ведет к соответствующему результату (например, применяет новое состояние ко всем выбранным элементам – если вернуться к примеру с полужирным начертанием, то весь текст становится полужирным).
- Убедитесь, что каждый контрольный элемент верно реагирует на правильный и неправильный ввод, включая граничные значения. К примеру, неверный ввод может выводить на экран сообщение, или подсвечивать элемент каким-то образом.
- Убедитесь, что диалоговое окно отображается и функционирует корректно:
    - При различных цветовых схемах
    - При различных настройках шрифтов
    - В режиме высокой контрастности
    - В режиме высокого DPI
- Убедитесь, что все изображения и другие медиафайлы локализованы.

Внешний вид диалоговых окон

Вы не закончили тестировать, не проверив следующее:

- Убедитесь, что команды меню, инициирующие открытие диалогового окна, заканчиваются многоточием (например, "Создать новый файл…"). Так принято в MS Windows. Если вы используете другую операционную систему, загляните в руководство по ее стилям.
- Убедитесь, что размер каждого элемента и расстояние между каждой парой элементов соответствует вашему руководству по стилю.
- Убедитесь, что размер диалогового окна соотносится с его контрольными элементами.
- Убедитесь, что элементы, находящиеся в неопределенном состоянии, выглядят так, как должны выглядеть согласно вашему руководству по стилю (обычно они становятся серыми или еще каким-то явным образом демонстрируют, что состояние не определено).
- Убедитесь, что образцы работы в диалоговом окне правильно отражают содержание и форматирование актуального документа. Если это не так, лучше вообще не показывайте их! Диалоговые окна, связанные с изменением форматирования, часто демонстрируют, к какому эффекту приведут внесенные изменения. Демонстрация актуального документа (или подходящей его части, например, выделенного в данный момент текста) увеличивает ценность предпросмотра работы функции. Если в вашем предпросмотре отображается какой-то абстрактный пример, который (возможно) чем-то похож на реальный документ – лучше вообще ничего не показывайте.

Конечно, эти мелочи выглядят как трата времени, но они тоже важны! Люди могут не обращать на них внимания сознательно, но эти детали влияют на то, как пользователи оценят качество вашего продукта не меньше, чем краши. Если неаккуратный вид приложения – первое, с чем сталкивается пользователь, это повлияет на его дальнейшее восприятие. Отполируйте все по-максимуму!

Ввод в текстовые поля

Вы не закончили тестировать, не покрыв своими тестами граничные значения для всех

текстовых полей приложения (не забудьте про комбо-боксы, которые можно редактировать).

- Null (при тестировании API)
- Пустое поле
- Один символ
- Два символа
- Несколько символов
- Много символов
- На 1 меньше максимального количества символов
- Максимальное количество символов
- На 1 больше максимального количества символов
- Пробелы
- Спецсимволы (запятая, нижнее подчеркивание) в тексте
- Пунктуационные символы
- Символы ASCII
- Символы из расширенной таблицы ASCII
- Немецкие символы
- Японские символы
- Символы иврита
- Арабские символы
- Символы Unicode из разных диапазонов
- Управляющие символы.

Обработка текста иногда прямо-таки кишит ошибками. Если ваше приложение на 100% в Unicode, считайте, что вам повезло. Но даже в этом случае попробуйте импортировать или экспортировать данные из кодировок, отличных от Unicode. Если приложение обрабатывает ASCII-текст – развлекитесь тестированием страниц в разных кодировках (попробуйте сменить кодировку во время ввода текста и посмотрите, что произойдет). А если приложение использует двухбайтовую или мультибайтовую кодировку – может, вам стоит задуматься о смене профессии.

Отмена и возврат

Вы не закончили тестировать, если вы не проверили работу отмены и возврата. Если ваше приложение не поддерживает отмену действий – вы соскочили, поздравляю. В противном случае вы должны сделать вот что:

- Подумать, должно ли отменяться любое пользовательское действие.
- Подумать, должна ли быть доступной возможность вернуть любое пользовательское действие.
- Протестировать один уровень отмены.
- Протестировать несколько уровней отмены.
- Протестировать один уровень возврата.
- Протестировать несколько уровней возврата.
- Воспользоваться возвратом большее количество раз, чем отменой. В некоторых приложениях "возврат" может означать "повтор действия".
- Протестировать смешанную последовательность отмены и возврата.

- Убедиться, что каждое отмененное и возвращенное действие правильно выводится в интерфейсе этих функций.
- Протестировать отмену и возврат при сохранении документов (некоторые приложения сбрасывают возможные для отмены и возврата действия после сохранения).
- Протестировать отмену и возврат, закрыв и переоткрыв документ.
- Протестировать отмену и возврат на разных билдах, если ваше приложение создает билды или использует другие билды (например, позволяет пользователю ссылаться на другие библиотеки). Содержание билда может измениться – как вернуть действие, субъект которого отсутствует в библиотеке?

Тестировать отмену и возврат вручную довольно просто, и зачастую вы найдете баги. Эти баги, как правило, легко исправляются. Самые интересные баги живут в комбинациях возвратов и отмен. Конечно, это можно проверять и вручную, но при помощи дрессированных мартышек (то есть автотестов) может получиться эффективнее.

Может, вы решите, что один человек будет тестировать отмену и возврат во всем приложении. Лично для меня лучше работает, если люди проверяют отмену и возврат по отдельности, каждый в своей области.

Печать

Вы не закончили тестировать, если не проверили, как ваше приложение выводит информацию на печать. Если помните, в старые "добрые" времена, до того, как операционные системы фактически стерли различия между принтерами, каждому приложению требовалось вступать с принтером в интимное знакомство, чтобы воспользоваться им. Теперь все иначе, и у вас больше времени поинтересоваться вот чем:

- Убедитесь, что смена ориентации страницы работает правильно. Попробуйте поменять ее для свежесозданного документа, для документа, с которым уже велась работа. Попробуйте изменить ориентацию напрямую через диалоговое окно приложения (например, через меню), и через диалоговое окно печати.
- Убедитесь, что печать на локальный принтер осуществляется.
- Убедитесь, что печать на сетевой принтер осуществляется.
- Убедитесь, что печать в файл работает. Все ОС, с которыми я знаком, позволяют печатать в файл вне зависимости от установленного принтера.
- Убедитесь, что печать на PCL-принтер работает. PCL родился, как язык управления для принтеров Hewlett-Packard, но в итоге стал практически стандартом.
- Убедитесь, что печать на PostScript-принтер работает. Этот язык управления был создан Adobe и тоже стал фактически стандартом. В нем легко разобраться, поэтому можно протестировать принтер, исследовав полученный файл, и сэкономив таким образом бумагу.
- Убедитесь, что печать в PDF работает. Существует множество бесплатных или очень дешевых редакторов, позволяющих это. Рассмотрите также возможность приобретения Adobe Acrobat, чтобы "официально" протестировать подобную печать.
- Убедитесь, что отмена печати, когда печать уже в процессе, работает. Мой нынешний принтер делает вид, что согласен с отменой, но продолжает плеваться страницами. Бесит!

- Убедитесь, что изменение настроек печати, поддерживаемых вашим приложением, дает желаемый эффект: например, количество копий, разбор по копиям, нумерация страниц.
- Убедитесь, что настройки, специфичные для принтера, работают. Они по идее должны соответствовать настройкам вашего приложения, но никогда не знаешь, где наткнешься на баг.

С первого взгляда кажется, что за большинство подобных проверок должны отвечать тестировщики операционной системы, но я часто сталкивался с мелкими изменениями, внесенными разработчиками в диалоговые окна печати. Хоть они и кажутся стандартными, но что-то в них не так. Такие изменения могут наплодить целые стада багов, именно потому, что разработчик уверен, что такая мелочь просто не может что-то сломать!

Даже когда диалоговое окно печати абсолютно, стопроцентно стандартное, я пробегаюсь по нему просто для самоуспокоения. То же самое касается настроек принтера. Да. все ДОЛЖНО работать нормально, но я буду счастливее, когда я ЗНАЮ, что оно так работает!

В общем случае это связано с оценкой рисков вашего приложения. Баги МОГУТ быть где угодно – где, с вашей точки зрения, вы их вероятнее всего найдете? Начинайте с этих областей, затем переходите к следующей группе риска, и так далее. Подключайте исследовательское тестирование, потому что баги любят кучковаться в местах, куда вы даже не подумаете заглянуть!

Особые режимы и состояния

Вы не закончили тестировать, если не проверили специальные режимы и состояния вашего приложения. В идеале стоило бы пробежаться по этому списку от и до, но я пока не встречал тестировщиков с таким количеством доступного времени. Как правило, выбирается один случай в качестве контекста, в котором будут проводиться все прочие тесты в этот день.

- Разные уровни масштабирования, если это применимо. Я сталкивался с приложением, для которого были написаны тысячи автотестов, и все было отлично. Потом кто-то попробовал изменить масштабирование и нашел целую кучу багов. Упс.
- Безопасный режим. Windows, стартующая в этом режиме, загружает только самое важное – базовый драйвер дисплея, голые сетевые стеки – и не запускает приложения и сервисы, находящиеся в автозапуске. Сможет ли ваш продукт выжить в таких условиях?
- Документы, находящиеся в совместном доступе для нескольких пользователей/компьютеров, одновременно и последовательно. Это особенно важно, если программа имеет доступ к базе данных (что произойдет, если кто-то одновременно с вами редактирует запись?), но если вы можете открывать документы из сетевых расположений или общих папок на локальной машине, кто-то еще может сделать то же самое с тем же документом, который вы редактируете.
- Файл не открыт, открыт несохраненный файл, открыт несохраненный файл с автосохранением, открыт сохраненный файл.
- Полноэкранный режим и другие варианты просмотра.
- Различные размеры окна приложения (и окна документа, если ваше приложение поддерживает интерфейс с несколькими документами), особенно размер по умолчанию при запуске, минимизированный, максимизированный, не максимальный, но растянутый до пределов экрана, очень маленький.

- Спровоцируйте режим ожидания, гибернацию и другие энергосберегающие режимы, работая с приложением. Через боль и отчаяние я узнал, что когда у приложения открыто модальное окно, это блокирует переход операционной системы в спящий режим.
- Выведите компьютер из различных режимов ожидания. Начатые до перехода в этот режим операции возобновились? Перезапустились? Или зависли?
- Изменение системных настроек. Поиграйте со скоростью мыши. Измените длительность повторного нажатия клавиш. Поменяйте системные цвета. Подхватит ли ваше приложение новые значения, когда запустится? Подхватит ли оно их, если оно уже запущено?
- Связывание и внедрение объектов (OLE). Если ваше приложение поддерживает OLE, вы попали в Диснейленд тестировщика! Правильно ли работает внедрение других OLE-объектов в ваше приложение? Как насчет встраивания документов из вашего приложения в другие, поддерживающие OLE? Правильно ли активируются и деактивируются внедренные приложения? Связанные OLE-документы изменятся, если изменен источник связи? как ваше приложение обращается со связями, если приложение, поддерживающее источник, более недоступно?
- Множественный выбор. Что произойдет, если вы измените форматирование текста при нескольких выделенных областях? А если нажмете "Вставить" в той же ситуации? Как должно работать?

Еще два специальных состояния не относятся к контексту прогона тестов. Это скорее дополнительные тесты, которые можно провести после других:

- Функция "отправить". Многие приложения имеют опцию отправки документа по электронной почте. Я видал случаи, когда попытка ей воспользоваться крашила приложение, или делала что-нибудь похуже.
- Вырезать, скопировать, вставить. "Сам в себя", в другой документ, в другие приложения, в документы, поддерживающие другие версии данных (например, копирование из Wordб вставка в текстовый редактор), в приложения, не поддерживающие никакую версию данных (что произойдет, если вы скопируете что-то из проводника и вставите в приложение) – надеюсь, вы уловили суть.

Международная доступность

Вы не закончили тестировать, если вы не убедились, что ваше приложение готово к использованию в различных странах. Даже если вы уверены, что оно никогда не будет использоваться за пределами вашей страны, стоит хотя бы одним глазком взглянуть на список ниже. Ваша команда может решить, что найденные в результате проблемы не заслуживают исправления, но вы хотя бы будете знать, где эти проблемы живут. И вы действительно на 100% уверены, что житель верхней Элбонии не захочет пользоваться вашем продуктом?

- Обратите внимание на культурно-специфичные изображения и термины. К примеру, красный цвет – сигнал опасности в ряде западных стран, а в других странах он обозначает счастье или удачу.
- Поисследуйте геополитические тонкости. К примеру, это могут быть карты, демонстрирующие спорные территории. Если ваше приложение отображает карты, готовьтесь к проблемам, как только оно выйдет за пределы вашей страны!
- Убедитесь, что ваше приложение поддерживает переключение системных языков,

языковых пакетов, и кодировок – как до запуска, так и после него.

- Убедитесь, что ваше приложение поддерживает изменение региональных настроек, как до запуска, так и после него: форматы времени и даты, символы и форматы валют, порядок сортировки, и т. д. Некоторые (или все) подобные настройки различаются для разных стран и языковых пакетов. Большинство операционных систем позволяют менять эти настройки отдельно от изменения языка или локали. В Windows это можно сделать через панель региональных настроек. К примеру, если ваше приложение работает с валютами, посмотрите, что будет, если вы измените символ валюты на "abc". Я встречал приложение, которому это совсем не понравилось!
- Убедитесь, что ваше приложение корректно работает с мультибайтными (например, японским), сложными (например, арабским) языками, и языками с записью справа налево (например, ивритом). Правильно ли перемещается по тексту курсор? Что произойдет, если смешать текст с написанием слева направо, и со "справа налево"?
- Убедитесь, что все контрольные элементы правильно взаимодействуют с IME. Это особенно важно, если вы планируете продавать ваш продукт в странах восточной Азии.
- Убедитесь, что ваше приложение умеет работать с различными раскладками клавиатуры. Региональные настройки (а также некоторые локали и языковые пакеты) применяют специфические раскладки клавиатур. Раскладку также можно менять и напрямую при необходимости.
- Убедитесь, что ваше приложение корректно работает с текстом ANSI, Unicode, а также мультибайтным текстом и нестандартными символами при вводе, отображении, редактировании и выводе.
- Убедитесь, что используется правильный порядок сортировки. Сортировка – это очень трудно! Спросите любого, кто хоть раз напоролся на распространенный баг с турецкой "i" при сортировке. Если вы опираетесь на правила сортировки операционной системы, то скорее всего, у вас все в порядке, но если в вашем приложении есть специфические виды сортировки – скорее всего, вы найдете баг.
- Убедитесь, что системная, пользовательская и постоянная локаль применяются так, как должны применяться. Используйте пользовательскую локаль, чтобы отображать данные для пользователя, системную – для работы с не-Unicode строками, и постоянную, чтобы форматировать данные для хранения.
- Убедитесь, что функции, зависимые от языка, работают нормально.
- Убедитесь, что ваши тест-кейсы учитывают эти проблемы. По моему опыту, тестировщики делают тут ошибку, аналогичную ошибкам разработчиков – не удивляйтесь, если разработчик укажет на баг в вашем кейсе!

Локализация

Тестирование международной доступности важно практически для любого приложения, но локализационное тестирование имеет смысл, только если вы переводите свое приложение на другие языки. Возможно, разницу между ними трудно запомнить, но она довольно простая: при тестировании международной доступности вы убеждаетесь, что приложение не конфликтует с другими локалями (например, не воспринимает в качестве десятичного разделителя только точку), а тестирование локализации направлено на возможность перевода вашего приложения на другие языки. Эти два вида тестирования похожи, но совершенно не зависят друг от друга.

Наиболее простой способ приступить к тестированию локализации – это использовать псевдолокализованный билд. Такой билд берет за основу исходный язык вашего приложения и псевдолокализует его, добавляя всякие интересные штуки в начале и конце каждой локализованной строчки ("интересные штуки" зависят от целевого языка, но могут включать, к примеру, двухбайтовые символы и символы, пишущиеся справа налево). Это сильно упростит ваше тестирование:

- Такой способ позволяет автоматизировать локализацию билда, а это дешевле и быстрее, чем локализация при помощи человека.
- Это позволяет протестировать локализованный билд, даже если вы не знаете языка, на который переведено приложение.
- Строки, которые должны переводиться, но не переведены, сразу бросаются в глаза – перед ними и после них нет добавленных символов.
- Строки, которые не должны переводиться, но переводятся, имеют спецсимволы в начале и конце, и их тоже легко обнаружить.
- Проще найти двухбайтовые баги.
- Проблемы интерфейса (обрезанные строки, верстка) легко обнаружить.

Если можете, проводите большую часть своего тестирования на подобных билдах. Это позволяет скомбинировать локализационное и функциональное тестирование. Тестирование билдов, которые по-настоящему локализованы, тоже очень важно, но куда проще. Если вы нашли серьезную проблему в локализованном билде, подумайте, как ее можно обнаружить на псевдолокализованном!

Помимо всего этого, есть вещи, которые стоит держать в уме, тестируя (надеюсь, что псевдо-) локализованный билд:

- Убедитесь, что все контрольные элементы интерфейса (не забудьте про диалоговые окна!) правильно ориентированы и масштабированы. Распространенные баги – это автомасштабированные элементы, не соответствующие друг другу по местоположению, и немасштабированные элементы, обрезающие содержание элемента.
- Убедитесь, что данные правильно упорядочиваются и сортируются.
- Убедитесь, что переход по TAB работает правильно (по идее, локализация не должна это затронуть, но я видал и более странные вещи).
- Убедитесь, что все строки, которые должны быть переведены, действительно переведены. Строка, которая не переведена, скорее всего, жестко закодирована.
- Убедитесь, что строки, которые не должны переводиться, не переведены.
- Убедитесь, что все сочетания клавиш переведены.
- Убедитесь, что каждое сочетание клавиш уникально.
- Убедитесь, что все горячие клавиши переведены.
- Убедитесь, что все комбинации горячих клавиш уникальны.

API

Если ваше приложение устанавливает в систему EXE, DLL, LIB или любые другие файлы (исчерпывающе описывает любые приложения, с которыми я сталкивался), вам нужно протестировать API. А может (по идее), не нужно – если только ваше приложение использует эти

DLL, или только один API – если EXE не поддерживает аргументы командной строки. Но, как знает любой тестировщик, "по идее" не всегда коррелирует с "на самом деле".

- Убедитесь, что все публично доступные API на самом деле доступны. Как вариант, проведите ревью кода. Альтернатива – инструменты, которые существуют на любом языке и проверяют именно это. В Microsoft мы использовали .Net Reflector. Для исполняемых файлов начните с вызова приложения через "-<command>', ":<command>", "/<command>", "\<command>" и "<command>"-аргументов командной строки, заменяя "command" на "?" или "help", или название файла. Если одна из вспомогательных команд работает, вы убедитесь, что приложение понимает аргументы командной строки, и узнаете, в каком именно формате оно их понимает.
- Убедитесь, что непубличные API не могут причинить вам вреда, если к ним получен "нелегальный" доступ. То, что API не публичен, не означает, что его нельзя вызвать. Языки управляемого кода обычно позволяют любому желающему получить доступ к непубличным методам и свойствам, а таблицы методов можно взломать. В большинстве случаев, конечно, те, кто так делают, не будут жаловаться, стреляя себе в ногу в результате, но убедитесь, что таким образом невозможно получить доступ к конфиденциальной информации. К примеру, просто скрыть от посторонних глаз ключ шифрования или код, проверяющий лицензию – недостаточная мера.
- Просмотрите все внешние и внутренние API, имеющие отношения к вашему продукту. Имеют ли они смысл? Соответствуют ли их уровни видимости желаемым? Понятно ли из их названия, что они делают и зачем?
- Убедитесь, что все публичные объекты, методы, свойства и процессы протестированы.
- Убедитесь, что все опциональные аргументы корректно работают, когда они определены и не определены.
- Убедитесь, что все возвращаемые значения и исключения верны и имеют ценность.
- Убедитесь, что все объекты, методы, свойства и процессы, ориентированные на многопоточность, действительно таковыми являются.
- Убедитесь, что каждый API можно использовать на всех поддерживаемых языках. К примеру, ActiveX должен управляться (как минимум) через C++, VB, VB.Net, C#.
- Убедитесь, что для всех публичных объектов, методов, свойств и процессов существует документация, и она верна. Удостоверьтесь, что образцы кода в документации компилируются и запускаются.

Платформа

Вы не закончили тестировать, если вы не подумали над платформами, которые нужно и не нужно включать в вашу тестовую матрицу. Список поддерживаемых платформ может варьировать в зависимости от контекста – пользовательское приложение и бизнес-приложение будут отличаться друг от друга в этом плане. Даже если вы официально поддерживаете ограниченный список платформ, полезно знать, что произойдет, если ваше приложение установят или запустят на каких-нибудь других. Платформы, о которых стоит подумать:

- Все поддерживаемые версии Windows, как минимум, Windows XP SP2, Windows XP SP последней версии, Windows Server 2003 SP последней версии, Windows Vista SP последней версии, Windows 7, 8 и 10.
- Apple OS X

- Ваша любимая сборка Linux
- Ваш любимый Unix
- 32-битная версия операционной системы при 32-битном железе.
- 32-битная версия операционной системы при 64-битном железе.
- 64-битная версия операционной системы при 64-битном железе.
- Различные SKU операционной системы.
- Совместимость различных SKU операционной системы.
- Совместимость различных операционных систем (к примеру, открытие файла через Windows-компьютер, если файл расположен в сетевом окружении Linux).
- Все поддерживаемые браузеры и их версии (IE, Opera, Firefox, Chrome).
- Работа с антивирусом и без.
- Работа с файерволом и без.

Просмотрите также требования Windows Logo. Если даже вы не собираетесь им соответствовать, или ваше приложение не предназначено для Windows, это неплохая отправная точка для новых тест-кейсов!

Конфигурации процессора

Вы не закончили тестировать, если не проверили различные конфигурации процессора. Частая проблема, с которой я сталкивался – это использование одной и той же марки, модели и конфигурации на компьютерах абсолютно всех разработчиков и тестировщиков. Это особенно верно для лабораторий. Да, это упрощает установку и решение проблем, но ваши пользователи живут в совершенно другом мире!

Это верно, даже если вы пишете приложение для организации, в которой требования к компьютерам контролируются и высечены в камне. Эта "стандартная" конфигурация будет регулярно меняться, и эти перемены займут месяцы и годы (после чего стандарт, конечно, снова изменится).

Поэтому убедитесь, что в вашей компании – у разработки, тестировщиков, менеджера проекта, и всех остальных, помогающих в создании продукта – имеются:

- Процессоры от различных производителей (Intel и AMD, если ваше приложение рассчитано на Windows).
- Несколько версий различных брэндов (к примеру, для Intel – мобильные и десктоп-Celeron, и Pentium)
- Однопроцессорные, двухпроцессорные, мультипроцессорные системы
- Одноядерные, двухъядерные, многоядерные процессоры.
- Процессоры с технологией гиперпотока (HyperThread) и без
- Полноформатные компьютеры и ноутбуки.
- 32-битные и 64-битные конфигурации.

Конфигурации железа

Вы не закончили тестировать, если не определились с конфигурациями железа, которые нужно включить или изъять из тестовой матрицы:

- Слабый компьютер
- Сильный компьютер
- Слабый ноутбук
- Мощный ноутбук
- Минимально допустимое разрешение экрана (в некоторых случаях это 640/480, но это зависит от ваших пользователей).
- Экстремально высокое разрешение экрана (да-да, воспользуйтесь этой отмазкой, пусть руководство купит этот огромный плоский экран!)
- Другие форм-факторы, относящиеся к делу
- Максимально мощная супер-машина
- Машина с минимальной конфигурацией
- Ноутбук с отключенными настройками управления питанием.
- Ноутбук с настройками управления питанием, выставленными на максимум.
- Ноутбук с настройками управления питанием, максимизирующими работу батареи.
- Конфигурации процессоров.

Конкретная разница между "сильными" и "слабыми" компьютерами зависит от приложения, вариантов использования, сценариев пользователя – минимальная конфигурация для профессиональной CAD будет сильно отличаться от обычной, нацеленной на массового пользователя программы для дизайна дома. Подумайте, какие платы, процессоры, производители должны быть охвачены. Возможно, полная матрица конфигураций выйдет такой огромной, что ваш бюджет ее не потянет!

Безопасность

Вы не закончили тестировать, если вы не подумали как следует про тестирование безопасности, и не приняли сознательного решения, что вы будете проверять и что нет. Раньше, когда даже корпоративные компьютеры редко были подключены к сети, тестирование безопасности было небольшой проблемой. Даже если компьютер хватал вирус, это не могло нанести большого вреда. Сейчас мы живем в мире, где вирусы и черви могут угробить почтовую систему всей корпорации, все мы получаем спам, и множество ни о чем не подозревающих людей живут с троянами в системе, которые неизвестно чем неизвестно ради чего занимаются. Тестирование безопасности стало очень важной задачей. Вот примеры тестов, о которых тут можно подумать:

- Покопайтесь в исходном коде, API и интерфейсе в поиске:
    - Проблем переполнения буфера
    - Проблем, связанных с атаками типа "Отказ в обслуживании"
    - Проблем, связанных с SQL-инъекциями
    - Вирусных атак
    - Нарушения конфиденциальности пользовательских данных (к примеру, включение данных пользователя в сохраненные файлы).
- Для MS Windows – воспользуйтесь Application Verified, чтобы убедиться, что приложение не создает NULL DACLS, а также для поиска других потенциальных проблем.
- Убедитесь, что уровень безопасности для ссылок и макросов достаточен, и безопасность обеспечивается.
- Убедитесь, что относительные именования файлов (например, "…\...\file") корректно

обрабатываются.

- Убедитесь, что временные файлы создаются в нужном месте и имеют правильные настройки прав доступа.
- Убедитесь, что ваше приложение корректно функционирует при использовании с различными ролями и правами пользователей.
- Убедитесь, что приложение правильно функционирует при сценарии частичного доверия.
- Убедитесь, что каждый ввод проверен на граничные значения.
- Убедитесь, что вы защитились от наиболее вероятных атак.

Даты и проблема-2000

Вы не закончили тестировать, если не проверили ваше приложение на отсутствие проблемы-2000. Несмотря на то, что мы давно уже пережили эту дату, проблемы, связанные с ней, продолжают возникать в приложениях. Если вы используете какую-либо форму Unix (к примеру, Apple-компьютер) – то ожидайте проблем в 2038 году, когда откажет 32-битная структура хранения времени. О, и раз уж вы занялись датами, поищите заодно и другие характерные для них дефекты – например, обработку високосного года.

- Убедитесь, что если год вводится двумя цифрами, то даты с 1 января 00 по 31 декабря 29 воспринимаются, как 2000-2029 годы соответственно.
- Убедитесь, что если год вводится двумя цифрами, то даты с 1 января 30 по 31 декабря 99 воспринимаются, как 1930-1999 годы соответственно.
- Убедитесь, что поддерживаются даты как минимум вплоть до 2035 года.
- Убедитесь, что даты високосных лет обрабатываются верно:
    - 29 февраля 1900 года не принимается
    - 29 февраля 1996 года принимается
    - 29 февраля 2000 года принимается
    - 31 декабря 2000 года принимается и идентифицируется как 366 день
    - 29 февраля 2001 года принимается
- Проверьте другие интересные даты, включая:
    - 31 декабря 1999
    - 1 января 2000 года непротиворечиво демонстрируется
    - 10 января 2000 года (первая семизначная дата)
    - 10 октября 2000 года (первая восьмизначная дата)
    - Запрет на ввод 13 месяца для 2000 года.

Производительность и стресс-тестирование

Вы не закончили тестировать, если вы не разбираетесь в характеристиках производительности вашего приложения и том, как оно ведет себя под нагрузкой. Тестирование производительности на первый взгляд достаточно прямолинейно: убедиться в том, что время выполнения операций вполне приемлемо, что тут может быть сложного? Сложность в симуляции достаточно реалистичных сценариев, и это особенно верно для стресс-тестирования! К примеру, вы тестируете сайт, от которого ожидается бешеная популярность. Как вы симулируете работу множества пользователей?

У меня нет простых ответов на эти вопросы, но тут можно рассмотреть ряд сценариев и условий:

Производительность

- Убедитесь, что тесты производительности существуют для каждого сценария, и выполняются регулярно.
- Убедитесь, что целевая производительность существует для каждого сценария, и приложение ей соответствует.
- Убедитесь, что тесты производительности проверяют правильные сценарии и точки ввода данных.
- Убедитесь, что оптимизация производительности достигает своей цели.
- Убедитесь, что производительность отвечает вашим задачам с включенными и отключенными настройками (например, анимацией меню).
- Сравните производительность с предыдущей версией.
- Сравните производительность с другими приложениями.

Стресс-тестирование

- Запустите приложение в условиях нехватки памяти
- Запустите его в условиях нехватки свободного места на диске
- Запустите в условиях, если свободная память закончилась, при помощи автоматизации
- Запустите в условиях, если память закончилась, в реалистичных сценариях (например, запустив множество других приложений, работающих с большим количеством документов)
- Запустите под нагрузкой из большого количества пользователей
- Запустите в сети, которая часто падает
- Запустите в высоконагруженной сети
- Запустите в сети с низкой пропускной способностью
- Запустите на машине с минимальной конфигурацией
- Откройте, сохраните, запустите с любых внешних носителей.

В процессе выполнения тестов следите за утечками памяти и других ресурсов.

Конфигурация и совместимость

Вы не закончили тестировать, если не обратили внимания на конфигурацию вашего приложения. Приложение может иметь множество конфигураций – глобальные и пользовательские конфигурационные файлы, глобальные и пользовательские настройки реестра, и т. п. Пользователи любят, когда у них есть возможность подстроить приложение под себя, чтобы оно выглядело и работало именно так, как им нужно. Они раздражаются, если при повторном запуске все эти настройки слетят. Совместимость с другими копиями того же приложения и другими приложениями им тоже важна: большинство пользователей рассчитывает на определенный уровень совместимости приложений между собой. Взаимодействия разных окон тоже попадают в эту категорию. Вот над чем можно поразмышлять, решив приступить к тестированию этих областей:

Конфигурация

- Убедитесь, что настройки, которые должны изменять поведение приложения, действительно это делают.

- Убедитесь, что настройки доступны/недоступны для администраторов, если это допустимо
- Убедитесь, что пользовательские настройки применяются к конкретному пользователю.
- Убедитесь, что ключи реестра верно установлены, а другие ключи реестра не модифицируются.
- Убедитесь, что пользовательские настройки не затирают глобальные.
- Подумайте о проблемах обратной совместимости, которые могут возникнуть из-за изменения или перемещения настроек и, как результат, нерабочей функциональности предыдущей версии. Как вариант, это может быть изменение значений по умолчанию.

Совместимость

- Убедитесь, что операции "вырезать", "скопировать" и "вставить" работают внутри вашего приложения.
- Убедитесь, что операции "вырезать", "скопировать" и "вставить" работают между вашим приложением и другими.
- Убедитесь, что перетаскивание работает внутри приложения.
- Убедитесь, что перетаскивание работает между приложениями.

Взаимодействия окон

- Убедитесь, что сортировка по глубине работает, обратите особое внимание на окна, которые должны отображаться поверх всех прочих (например, окно помощи).
- Убедитесь, что все модальные диалоговые окна блокируют доступ к другим частям приложения, а окна, не являющиеся модальными, доступ позволяют.
- Убедитесь, что фокус корректно наводится на окна и диалоговые окна.
- Убедитесь, что размер окна верно восстанавливается после минимизации и максимизации.
- Убедитесь, что размер окна верен при первом запуске.
- Убедитесь, что размер окна сохраняется после того, как он был изменен вручную.
- Убедитесь, что сценарии, работающие с несколькими окнами, выполняются верно.
- Убедитесь, что команды упорядочивания окон (например, каскадом) работают верно.
- Убедитесь, что несколько копий вашего приложения верно работают в любых ситуациях (к примеру, что модальное диалоговое окно одной из копий не блокирует все остальные).

Установка

Вы не закончили тестировать, если вы не проверили процесс установки вашего приложения при условиях, описанных ниже. Некоторые из них специфичны для Windows, но у других операционных систем будет нечто похожее.

- Установка с CD-ROM/DVD-ROM
- Установка из сетевого размещения.
- Установка с локального диска.
- Установка в сетевом размещении.
- Установка из рекламного инсталлятора, когда иконки и другие точки запуска приложения создаются (то есть оно как бы рекламируется пользователю), но фактическая установка происходит только после первичного запуска приложения. Этот метод также известен как "установка по запросу" или "установка при первом использовании".

- Установка, не требующая вмешательства, названная так из-за отсутствия необходимости подтверждать что-то в системных сообщениях. Довольно сложный тест, так как механизм установки, встроенный в ОС, поддерживает множество опций командной строки, а ваше приложение может поддерживать еще больше. О, счастье комбинаторного тестирования!
- Массовая установка через корпоративный деплой, например, Microsoft Systems Management Server.
- Обновление с предыдущих версий. Это может стать довольно сложной задачей в зависимости от того, сколько версий вашего приложения уже вышло, и для каких из них вы поддерживаете обновления. Если все ваши пользователи всегда обновляются после выхода новой версии, можно считать, что у вас все хорошо. Но если в мире существуют пользователи, не обновлявшиеся в течение пяти-шести версий – и добросим сюда различные сервис-паки и хотфиксы – ваша задача усложняется!
- Деинсталляция. Убедитесь, что удаляются не только файлы приложения и общие файлы, но и значения в регистре и другие конфигурационные изменения. Удостоверьтесь, что компоненты, находившиеся в совместном доступе с другими приложениями не удалены (или удалены, зависит от того, существуют ли еще приложения, нуждающиеся в них). Попробуйте деинсталлировать приложения в разном порядке: установите приложение A, затем приложение B, затем удалите приложение A, а после него – приложение B.
- Повторная инсталляция после деинсталляции новой и предыдущих версий вашего приложения.
- Установка на все поддерживаемые ОС и SKU. Возможно, для Windows придется вспомнить аж про Windows 95 – для Linux определите список поддерживаемых дистрибутивов.
- Минимальная, обычная, полная и особая установка. Убедитесь, что каждый тип установки устанавливает нужные файлы, обеспечивает требуемую функциональность, и правильно устанавливает значения регистра и конфигурации. Попробуйте проапгрейдить или даунгрейдить тип установки – от минимума к полной, например, или удалить какую-нибудь функцию. Убедитесь, что добавлены или удалены верные файлы, и соответствующая функциональность появляется или исчезает.
- Установите компоненты локально, с запуском из сети, с установкой при первом использовании, и недоступные. В зависимости от способа создания установщика, установка может позволять отдельным компонентам устанавливаться локально, или запускаться из сети, или устанавливаться по запросу, или не устанавливаться вообще. Убедитесь, что каждый компонент поддерживает соответствующий тип установки – ядро вашего приложения вряд ли должно поддерживать вариант "Не устанавливать".
- Проверьте компоненты, устанавливающиеся при первом запуске. Убедитесь, что они устанавливаются тогда, когда должны (а не раньше), и что они устанавливаются в правильном месте (что произойдет, если папка назначения удалена?) и правильно регистрируются.
- Проверьте компоненты, запускающиеся из сети. Убедитесь, что приложение вообще стартует – некоторые приложения при таком раскладе даже не запустятся, особенно если сетевое расположение дает права только на чтение. Что произойдет, если сеть недоступна, когда вы пытаетесь запустить приложение? Что произойдет при обрыве сети, когда приложение уже запущено?
- Убедитесь, что установка в папки глубокой вложенности работает правильно.
- Убедитесь, что проверки, которые проводит установщик (например, достаточность места на диске) работают нормально.

- Убедитесь, что ошибки, которые должен выдавать установщик (например, при нехватке места на диске) появляются в нужный момент.
- Убедитесь, что "обычные" пользователи и пользователи с ограниченным доступом (то есть не являющиеся администраторами), могут запустить приложение, если оно было установлено при помощи администраторских прав. В этом случае особенно вероятны проблемы при сценарии "Установка по запросу".
- Убедитесь, что приложение корректно работает при удаленном (например, через Microsoft Remote Desktop/Terminal Server) и виртуальном (Microsoft Virtual PC/Virtual Server) доступе. Графические приложения обычно плохо справляются с такими ситуациями. Я видел случаи, когда приложения просто отказывались загружаться, если их запускали через Terminal Server.
- Выполните "обычную" установку, а затем попробуйте ее изменить, добавив дополнительные функции.
- Выполните "специальную" установку, а затем попробуйте ее изменить, убрав функции.
- Выполните "обычную" установку, удалите один-два установленных файла, затем попробуйте починить приложение.
- Выполните "специальную" установку, включающую нетипичные функции, удалите один или несколько установленных файлов, затем попробуйте починить приложение.
- Пропатчите предыдущие версии. Накат патчей отличается от обновления – обновление обычно заменяет все установленные файлы, а патч перезаписывает только некоторые из них.
- Выполните небольшое обновление версии, которая до этого была пропатчена.
- Примените патч к уже обновленной версии.
- Обновите приложение, которое модифицировалось после установки.
- Пропатчите приложение, которое модифицировалось после установки.

Установка: особые случаи

Выше перечислены стандартные сценарии. Рассмотрите также ряд специальных условий. Как и в предыдущей части – хоть некоторые из этих проверок и специфичны для Windows, у других операционных систем есть аналоги для них.

Локальное кэширование

Инсталлятор может разрешать хранение установочных файлов в кэше на локальном диске с целью ускорения починки и других операций настройки приложения.

- Убедитесь, что кэшируются нужные файлы/архивы.
- Убедитесь, что все файлы, находящиеся в совместном доступе с другой функцией или другим приложением, правильно обрабатываются при установке, удалении и повторной установке.
- Убедитесь, что установка нескольких программ и нескольких версий одной программы приводит к правильному разделению кэша между ними. Это особенно критично для файлов, находящихся в совместном доступе – представьте, какой ужас начнется, если удаление одного приложения приведет к удалению файла, которым пользуются другие программы!
- Убедитесь, что удаление файла или архива из кэша приводит к переустановке

приложения, и файл/архив снова появляются в кэше.

Проверка инсталлятора

- Убедитесь, что все возможные пути через установку (включая отмену в любой точке) правильно работают.
- Убедитесь, что установщик включает в себя правильные компоненты, файлы, и настройки реестра.
- Убедитесь, что любые особые действия или условия, создание ярлыков, и другие случаи правильно работают.
- Убедитесь, что доступны нужные типы установки.
- Убедитесь, что отмена уже стартовавшей установки на самом деле отменяет ее, не оставляя в системе следов неоконченной работы.

Установка для нескольких пользователей

Что будет, если множество пользователей начнут копаться в конфигурации установки вашего приложения?

- Убедитесь, что приложение будет правильно работать для пользователя 2 после того, как пользователь 1 установит/изменит/повредит его.
- Убедитесь, что пользовательские функции установлены для пользователя 2, даже если по факту устанавливал их пользователь 1.
- Убедитесь, что пользовательские настройки пользователя 2 не меняются, если аналогичные настройки меняет пользователь 1.

Сетевая установка

Можете ли вы установить свое приложение из сети, а не из локального расположения?

- Убедитесь, что деинсталляция проходит чисто и правильно, если приложение было установлено из сети.
- Убедитесь, что нужные файлы устанавливаются в сеть и локально, если приложение установлено как "запускающееся из сети".

Обновления

Вы не закончили тестировать, если вы не разобрались, как ваше приложение чувствует себя при установке поверх предыдущих версий, а также при обновлении операционной системы. Может, стоит также протестировать установку предыдущей версии поверх более новой, или одновременно с новой. Подумайте, нужно ли вам проверять все три комбинации: обновление только вашего приложения, обновление только операционной системы, обновление ОС и приложения одновременно.

Обновление приложения

- Убедитесь, что обновление поверх предыдущей версии заменяет соответствующие файлы и не заменяет ничего, что не должно.
- Убедитесь, что новая версия и предыдущие версии приложения могут сосуществовать в системе.
- Убедитесь, что нужные файлы существуют/не существуют после обновления, и что их версии верны.

- Убедитесь, что настройки по умолчанию выставлены верно.
- Убедитесь, что предыдущие настройки и файлы сохранились/изменены.
- Убедитесь, что вся функциональность правильно работает, если предыдущая версия и/или новая версия установлены с запуском из сети.
- Убедитесь, что любые функции и приложения, зависящие от файлов или функциональности, которая затронута обновлением, работают правильно.

Обновление операционной системы

- Убедитесь, что обновление с предыдущей версии заменит только нужные файлы.
- Убедитесь, что функциональность корректно работает.
- Убедитесь, что любые функции и приложения, зависящие от файлов операционной системы или затронутой обновлением функциональности, работают правильно.

Документация

Вы не закончили тестировать, если вы не просмотрели всю документацию – чтобы убедиться, что она верна, и почерпнуть идеи для тест-кейсов. Случаи, когда образцы кода, размещенные во вспомогательных материалах, не компилировались по той или иной причине, не поддаются подсчету. Я видел документацию, в которой был изображен неактуальный интерфейс, и видел интерфейс приложения, ни слова про который не было сказано в документации. Подумайте еще о том, чтобы просмотреть запросы в поддержку, касающиеся предыдущей версии вашего приложения. Давно ли вы видели исходный код? Код-ревью – это простой способ выявить то, что должно было быть временным (вроде сообщений или функций), но вот-вот отправится в релиз и удивит покупателя. Перечислять можно бесконечно.

- Просмотрите все отложенные и не исправленные по другим причинам баги предыдущих релизов.
- Просмотрите запросы в поддержку от предыдущих релизов.
- Просмотрите отчеты об ошибках, переданные пользователями предыдущих релизов.
- Убедитесь, что все сообщения об ошибках в вашем приложении точны и удобопонятны.
- Убедитесь, что текст ошибок ввода точно указывает на суть проблемы.
- Убедитесь, что ваши обучающие материалы верны: шаги правильные, интерфейс соответствует актуальному, и так далее.
- Просмотрите все разделы помощи на предмет ее технической точности.
- Убедитесь, что все контекстные подсказки верны.
- Убедитесь, что все образцы кода правильно функционируют.
- Убедитесь, что все образцы кода соответствуют стандартам программирования.
- Просмотрите исходный код на предмет:
    - Правильности
    - Строк кода, которые не покрыты кейсами
    - Проблем безопасности (см. чек-лист для безопасности)
    - Потенциальных утечек памяти.
    - Мертвого кода
    - Правильной обработки ошибок.
    - Вызова устаревших и заблокированных функций.
    - Соответствия стандартам и гайдлайнам.

    - Недопустимых сообщений, видимых пользователю.
- Убедитесь, что вы обсудили новую функциональность и целевую аудиторию с командой.
- Убедитесь, что вы спросили у разработчика, на какие места нужно обратить особое внимание.
- Убедитесь, что вы обсудили новую функциональность со службой поддержки.
- Убедитесь, что вы проработали свои тесты совместно с командой разработки и тестирования.
- Убедитесь, что вы обсудили возможные кросс-функциональные трудности с командой.
- Убедитесь, что вы завершили функциональное тестирование.
- Убедитесь, что вы завершили кросс-функциональное интеграционное тестирование.
- Убедитесь, что вы протестировали продукт в условиях реального использования и так, как с ним бы обращался ваш пользователь.