Министерство образования Республики Беларусь
Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий
Кафедра технологий программирования

Методические указания
к выполнению лабораторной работы №7

по дисциплине «**Надёжность программного обеспечения**»
для специальности 1-40 01 01 Программное обеспечение информационных технологий

на тему «**Тестирование standalone-приложений**»

Новополоцк 2017 г.

Название*:* «Тестирование standalone-приложений».

Цель работы: научиться правильно подходить к тестированию standalone-приложений, эффективно справляться с обнаружением дефектов.

Теоретическая часть

Тестирование standalone-приложений почти не отличается от тестирования WEB, за исключением некоторых нюансов:

– безопасность таких приложений напрямую зависит от шифрования файлов данных, составляющих приложение. Такие файлы не хранят в открытом виде. Для проверки безопасности таких файлов следует просто проверить файлы на наличие смысловой нагрузки или данных, которые могут способствовать взлому;

– кроссплатформенное тестирование приложений играет важную роль. Не факт, что приложение поведет себя одинаково на разных версиях Windows, не говоря уже о проверке на Unix платформах, если, конечно, приложение собрано под такую платформу. Такие нюансы проверяются обычно на уровне AT (Full Acceptance Test), так как зачастую дефекты проявляются в интерфейсе программы. Всё, что нужно сделать на этом этапе проверки, это попросту запустить приложение на разных платформах/версиях;

– интерфейс состоит из тех же кнопок, тулбаров, поп-ап списков, label-ов, numeric-up-down элементов и т.д. Здесь тестирование сводится к обычному чек-листу – тем проверкам, которые вы сами для себя составили, как план. Допустим кнопка: должна нажиматься, выполнять какие-то действия после нажатия, явно показывать, что она нажата, или не активна. На этом этапе тестирования следует каждый элемент привести в действие и просмотреть, чтобы ни одна деталь интерфейса не ускользнула из вида;

– X86/X64. На разрядность приложения следует также обратить внимание, учитывая то, что не каждое 32-битное приложение корректно запустится на 64-битной системе и ни одно 64-битное приложение не будет работать на 32-битной системе;

– тестирование производительности так же играет важную роль. Приложение не должно забирать все ресурсы данного стенда, а его требования не должны превышать требования, описанные в ТЗ. Проверка производительности делается просто: открывается диспетчер задач, выбираются процессы, создаваемые данным приложением, совершается определенное действие в приложении, наблюдается активность процессов приложения. Банальные действия не должны забирать по 5 мегабайт системы. Если абстрагироваться, то есть еще несколько вещей, которые полезно знать при тестировании таких приложений:

– клиент-серверные приложения. В таких приложениях важно следить за тем, чтобы данные не передавались в открытом виде. Обычно это можно делать в режиме отладки приложения, при тестировании в режиме «белого ящика», то есть приложение с открытым исходным кодом;

– install/uninstall. Естественно, что эти операции так же создаются разработчиками и там тоже не может быть всё гладко. Следует обратить внимание на интерфейс, а также на то, что приложение удаляется полностью и не висит в трее или оставляет за собой мусор;

– compatibility testing. Тестирование совместимости с различными платформами. Наблюдаем, как ведет себя приложение на разных версиях операционки и, в общем-то, и все;

– upgrade testing. Проверяем работоспособность приложения после обновления/пропатчивания и т.д.

Пример:

Дано приложение – графический редактор Adobe Photoshop. Проведем полное тестирование данного standalone-приложения.

Будет уместным (и правильным) для начала провести Smoke Test. Проверяем основные функции программы, запускаемость, наличие критических ошибок, препятствующих нормальной работе приложения.

Далее следует провести тест GUI на наличие внешних дефектов интерфейса программы. Не выровненные по одному краю контролы, проблемы с отображением графических элементов, наличие несоответствий в стиле некоторых элементов.

Следом проведём функциональное тестирование программы. Так как это графический редактор, следует проверить инструменты данного приложения, такие как фильтры, коррекция цвета, утилиты работы с изображением и т.д.

В Adobe Photoshop можно провести Upgrade Test, так как приложение способно обновляться и пропатчиваться. Суть теста в том, чтобы приложение работало корректно после установки обновлений.

Оценка производительности – очень важный элемент тестирования для подобных приложений. Суть проверки описана в теоретической части.

Проводить проверку на лицензирование не имеет смысла, если у вас не активированная/пиратская версия приложения.

После всех этих кейсов, следует удалить приложение и проследить, чтобы оно не оставляло следов в системе (реестр, документы, системный диск).

Практическая часть

Содержание задания

Провести полное тестирование приложения из полученного варианта. Выполнить **не менее трёх** проверок по каждому типу тестов (smoke тест, функциональный, GUI, установка и удаление, производительность, кроссплатформенность, совместимость, обновление).

Варианты заданий

1 Adobe Reader.
2 AIMP.
3 Google Chrome.
4 Internet Explorer.
5 KMPlayer.
6 Macromedia Flash.
7 MathCad.
8 Microsoft Excel.
9 Microsoft PowerPoint.
10 Microsoft Word.
11 Mozilla Firefox.
12 Notepad++.
13 Opera.
14 Paint.
15 Skype.
16 Torrent-клиент.
17 Total Commander.
18 Viber.
19 Visual Studio.
20 WinRAR.
21 Блокнот.
22 Игра (любая игра, имеющаяся на вашем компьютере, либо телефоне).
23 Калькулятор.
24 Антивирус (любой антивирус, с которым знакомы).
25 Telegram Messenger.
26 Media Player Classic (K-Lite).
27 CCleaner.
28 Dropbox-клиент.
29 VirtualBox.
30 Foxit Reader.

Можно предложить индивидуальный вариант задания (необходимо согласовать с преподавателем)!

Порядок выполнения работы

1 Ознакомиться с теоретической частью.
2 Получить индивидуальное задание у преподавателя.
3 Установить приложение, из полученного варианта.
4 Провести Smoke Test.
5 Провести функциональные тесты.
6 Провести GUI тесты.
7 Проверить установку/удаление.
8 Провести тестирование производительности.
9 Провести кроссплатформенное тестирование.
10 Провести тестирование совместимости.
11 Провести Upgrade Test.
12 Составить отчёт о проделанной работе.
13 Показать отчёт преподавателю.
14 После защиты, выполненного задания, скинуть отчёт преподавателю (в **названии** указать **фамилию и номер лабораторной**, например, Ivanov7).

Порядок выполнения пунктов не обязательно совпадает с ходом вашего выполнения задания. Некоторые тесты можно опустить по причине невозможности их проведения (в отчёте пояснить, по какой причине тесты не проводились).

Содержание отчета

1 Титульный лист.
2 Цель работы.
3 Скриншот приложения, которое тестировали.
4 Описание логики приложения.
5 Описание трёх проверок, для описанного приложения по каждому из типов тестирования.
6 Отчёт по качеству приложения, из варианта задания.
7 Вывод о проделанной работе.
Защита работ проводится индивидуально.

Контрольные вопросы

1 Основные отличия тестирования Standalone от WEB-приложений.
2 Что такое Upgrade Test?
3 Что такое Compatibility Test?
4 На каком уровне тестирования проверяют интерфейс приложения?