

ПОВЫШЕНИЕ КАЧЕСТВА ВЕБ-ОРИЕНТИРОВАННЫХ ПРИЛОЖЕНИЙ НА СТАДИИ РАЗРАБОТКИ

С.С. Куликов

Белорусский государственный университет информатики и электроники,
кафедра программного обеспечения информационных технологий
ул. П. Бровки, 6, г. Минск, РБ
телефон: +37517-2938881; e-mail: kulikov@bsuir.by
web: www.bsuir.by

В докладе представлен комбинированный (с точки зрения разработчика и специалиста по обеспечению качества) взгляд на принципы и технологии повышения качества веб-ориентированных приложений на стадии разработки. Основное внимание уделено вопросам обеспечения качества таких элементов веб-ориентированных приложений как формы и ссылки. Рассмотрены вопросы инсталляционного тестирования веб-ориентированных приложений.

Веб-ориентированные приложения, инсталляционное тестирование, качество, ссылки, формы.

1 ИНСТАЛЛЯЦИОННОЕ ТЕСТИРОВАНИЕ

Инсталляционное тестирование – тестирование, направленное на выявление дефектов программного обеспечения, влияющих на протекание стадии инсталляции (установки) и деинсталляции (удаления) приложения.

При разработке веб-ориентированного программного обеспечения и планировании его тестирования следует уделить особое внимание вопросам контроля путей и процесса инсталляции.

Пути. Ещё на стадии разработки следует принять в качестве стандарта использование только относительных путей с префиксами, позволяющими превратить их в абсолютные. Префиксы следует задавать в процессе инсталляции (хорошим решением будет попытка приложения определить их самостоятельно на основе анализа переменных среды выполнения).

При формировании путей следует придерживаться правила об использовании только нижнего регистра символов, что позволяет исключить ошибки в приложении, связанные с переходом на регистрочувствительные файловые системы.

Быстро проверить корректность путей на стадии тестирования помогает анализ на наличие повреждённых ссылок (вызывающих ответ веб-сервера о возникновении ошибки 404).

Инсталлятор. Данный компонент веб-ориентированного приложения (являющийся обязательным для сложных приложений или приложений, ориентированных на широкий круг пользователей) должен выполнять следующие проверки (таблица 1).

ТАБЛИЦА 1

Необходимые проверки

№	Суть проверки
1	Наличие и контрольную сумму всех входящих в инсталляционный комплект файлов.
2	Наличие свободного места на диске (перед каждой операцией, приводящей к его уменьшению).
3	Достаточность имеющегося в наличии свободного места для полноценного функционирования приложения на протяжении некоторого разумного периода времени.
4	Ограничения доступного объёма оперативной памяти и максимального времени выполнения.
5	Версию серверной ОС, сервера приложений и веб-сервера.
6	Доступность и версии расширений сервера приложений.
7	Версию СУБД, возможность подключения (корректность имени пользователя, пароля, наличие используемой БД).
8	Возможность установки исходящих соединений (если они необходимы приложению).
9	Значение критических для безопасности, производительности и общей работоспособности приложения параметров сервера приложений и ОС.

Если разрабатываемый проект предполагает установку и настройку приложения разработчиком, применение модульного тестирования значительно сокращает время диагностики и исправления ошибок [1].

Следует учесть, что в процессе инсталляции приложения могут реализовываться различные варианты его установки (таблица 2), вероятность возникновения каждой из которых тем больше, чем для более широкого набора задач используется приложение.

ТАБЛИЦА 2

Варианты установки приложения

№	Описание
1	Новая среда исполнения, в которой приложение ранее не было инсталлировано.

- 2 Обновление существующей версии приложения.
- 3 Изменение текущей версии на более старую.
- 4 Повторная установка приложения с целью устранения возникших проблем.
- 5 Повторный запуск инсталляции (после отмены или сбоя).
- 6 Удаление приложения.
- 7 Установка нового (дополнительного) приложения из семейства приложений.

Рассмотрим обобщённый чек-лист (набор идей тестов) инсталляционного тестирования веб-ориентированного приложения (таблица 3).

ТАБЛИЦА 3

Чек-лист инсталлятора

№	Идея теста
1	Позволяет ли инсталлятор указывать пути размещения файлов, данные для соединения с СУБД, имя БД и тому подобную информацию.
2	Проверяет ли инсталлятор наличие и целостность всех необходимых файлов перед началом инсталляции.
3	Проверяет ли инсталлятор наличие свободного места на диске для установки приложения.
4	Проверяет ли инсталлятор наличие свободного места на диске для устойчивой работы приложения.
5	Проверяет ли инсталлятор настройки среды исполнения, критичные для работы приложения.
6	Проверяет ли инсталлятор версии серверного ПО и его настройки, критичные для безопасности, производительности и общей работоспособности приложения.
7	Проверяет ли инсталлятор возможность установки исходящих сетевых соединений.
8	Показывает ли инсталлятор, какие шаги он собирается выполнить, выполняет, выполнил (с указанием результата выполнения).
9	Дает ли инсталлятор рекомендации по устранению обнаруженных ошибок.
10	Позволяет ли инсталлятор вернуться на предыдущий шаг (шаги) инсталляции, внести изменения в параметры и продолжить инсталляцию с новыми параметрами.
11	Учитывает ли инсталлятор варианты установки приложения, перечисленные в таблице 2.

Применение такого набора тестов позволяет предусмотреть возможные проблемы стадии инсталляции приложения, а также реализовать на стадии разработки способы решения возможных проблем данного класса.

2 ФОРМЫ

Формы в веб-ориентированных приложениях в общем случае делятся на два основных вида (таблица 4).

ТАБЛИЦА 4

Виды веб-форм

Название	Описание
Одно- оконные формы	Полностью (со всеми своими полями) расположены на одной странице
Пошаго- вые фор- мы	Новые поля появляются (после загрузки новой страницы или в рамках старой с использованием JavaScript/AJAX) по мере заполнения уже показанных

Рассмотрим набор ключевых областей разработки и тестирования форм с точки зрения того, что следует и не следует делать.

Путь (URL). После обработки данных формы пользователь приложения должен попадать на такую страницу, которая несёт полезную информацию или минимизирует усилия пользователя по выполнению последующих действий. Так, например, если пользователь хочет изменить количество заказанных товаров в корзине интернет-магазина, он меняет число в соответствующем поле, отправляет форму и должен оказаться снова на странице корзины.

Недопустимыми являются ситуации, когда после отправки формы пользователь оказывается на странице, не позволяющей ему определить результат выполнения отправки данных или не содержащей части приложения, с которой пользователю необходимо продолжить работу.

Работа с веб-ориентированным приложением должна быть похожа на работу с классическим настольным приложением в плане своей последовательности и плавности. Следует избегать лишних сообщений, действий и т.п. Способ реализации этого требования следует продумывать на стадии проектирования приложения и проверять в процессе тестирования.

Месторасположение. Если форма расположена достаточно далеко от верхнего края страницы, следует обеспечить автоматическое позиционирование элемента страницы так, чтобы пользователю не приходилось применять прокрутку для доступа к форме.

Общий вывод по управлению путями и месторасположением форм таков: для перехода к некоторой операции или следующему шагу некоторой операции пользователю должно быть достаточно выполнения одного действия.

Ошибочные ситуации. Если пользователь ввёл некоторые данные некорректно (или не ввёл вообще), форма должна быть показана заново, и при этом все введенные данные (за исключением паролей и полей CAPTCHA) должны сохранять свои значения. Также в удобном для восприятия месте (как правило, сразу над формой) следует недвусмысленно указать причину неудачи отправки данных. Все неверно заполненные поля следует визуально выделить, а рядом с полем (при возможности) указать суть ошибки заполнения и подсказку по правильному заполнению.

АБЛИЦА 4

Ситуация, когда значения полей устанавливаются в состояние по умолчанию, не может считаться допустимой. Также недопустимым является отсутствие пояснения причины неудачи отправки данных.

Вывод: удобство использования приложения можно значительно повысить, увеличив информативность сообщений о неверных действиях пользователя, а также минимизировав его усилия по устранению ошибок.

Поля и их значения. Основное правило относительно полей форм, следование которому позволяет избежать недостатков удобства использования веб-ориентированного приложения и упростить его функциональную часть, может быть выражено так: в одном поле формы должно располагаться одно значение.

Так, например, разделение поля «Дата» на поля «Год», «Месяц» и «День» позволяет исключить необходимость анализа введенных данных на предмет соблюдения формата, а также понижает возможность того, что пользователь перепутает, например, месяц и день [2].

Удобство использования приложения позволяет повысить использование значений по умолчанию и размещение рядом с полями формы примеров заполнения. Однако категорически не рекомендуется приводить примеры паролей, т.к. низкоквалифицированные пользователи склонны использовать готовый пример, что представляет угрозу безопасности приложения. Даже уникальный пароль, генерируемый каждый раз заново, приводить в качестве примера не рекомендуется: он может попасть в кэш браузера, откуда его может извлечь злоумышленник.

Значения полей и спецсимволы. Разработчикам и специалистам по обеспечению качества веб-ориентированных приложений следует помнить, что через поле, теоретически, могут быть переданы любые символы. Из понимания этого факта следует необходимость тщательной фильтрации данных перед передачей их в базу данных или иной приемник.

Фильтровать данные и правильно оформлять поля формы также следует в силу того факта, что в случае неверного заполнения части полей данные будут снова показаны в форме. Если введенные данные содержат в себе ключевые для HTML последовательности, интерфейс пользователя может оказаться искаженным самым непредсказуемым образом.

JavaScript в формах. JavaScript, AJAX и сопутствующие технологии являются хорошими инструментами, значительно повышающими интерактивность и производительность интерфейса пользователя веб-ориентированных приложений. Так, например, можно проверять значения полей формы на корректность ещё в процессе заполнения или перед отправкой данных из формы. Можно выводить эстетичные информативные подсказки или сообщения об ошибках.

Однако следует помнить, что клиентское ПО пользователя может не поддерживать JavaScript (в силу технических ограничений или настроек) или же код, написанный на JavaScript, может выполняться некорректно в том или

ином браузере.

В таком случае форма, рассчитанная на активное использование JavaScript и лишённая альтернативных способов взаимодействия с пользователем, становится как минимум неинформативной, а как максимум – неработоспособной.

Вывод: на стадии проектирования следует предусмотреть варианты использования формы без применения JavaScript.

Пошаговые формы. Основная сложность работы с пошаговыми формами заключается в неочевидности их функционирования при возврате на предыдущие шаги или восстановлении работы после обрыва сетевого соединения. Эта проблема требует отдельного решения в каждой конкретной ситуации, однако существуют и общие рекомендации.

Следует однозначно давать понять пользователю, разрешено ли ему возвращаться на предыдущие шаги заполнения формы. Если разрешено – соответствующий механизм должен быть тщательно продуман с целью исключения возможности искажения и потери данных. Если запрещено – должен быть реализован механизм, явно запрещающий возврат, а в случае попыток совершить таковой должно появляться сообщение об ошибке. Также, возможно, следует предпринять дальнейшие меры (например, перенаправить пользователя на первый шаг формы для повторного заполнения).

Вывод: пошаговые формы сложнее однооконных, а потому их разработке и тестированию следует уделять повышенное внимание.

В завершение рассмотрения способов повышения качества веб-ориентированных приложений в контексте обработки форм рассмотрим обобщенный чек-лист (таблица 5).

ТАБЛИЦА 5

Обобщенный чек-лист для форм

№	Следует проверить
1	Расположение формы на экране перед заполнением, после неверного заполнения, после верного заполнения.
2	Сохранение или изменение URL в адресной строке браузера при отправке данных из формы.
3	Реакцию приложения на отправку формы с незаполненными обязательными полями.
4	Реакцию приложения на отправку формы с неверно заполненными полями (числа вне диапазонов, строки превышают допустимую длину и т.п.)
5	Реакцию приложения на отправку формы с противоречащими здравому смыслу данными (например, дата рождения – в будущем).
6	Реакцию приложения на отправку формы с полями, содержащими спецсимволы.
7	Восстановление значений полей после отправ-

- ки формы с неверно заполненными полями.
- 8 Информативность сообщений об ошибках.
- 9 Функциональную сгруппированность полей формы. Информативность надписей, подсказок.
- 10 Работоспособность и удобство использования формы с отключённым JavaScript.

3 ССЫЛКИ

Ссылки являются основой функционирования любого веб-ориентированного приложения, а потому любые недоработки, связанные со ссылками, приводят к возникновению серьёзных ошибок [3]. Ниже предлагается набор общих правил разработки и тестирования веб-ориентированных приложений в контексте предотвращения возникновения проблем со ссылками.

Первая проверка ссылки должна следовать сразу за добавлением ссылки. Вне зависимости от того, какую функциональную нагрузку несёт ссылка, её корректность может и должна быть проверена непосредственно при добавлении в приложение. В противном случае диагностика ошибки, связанной с данной ссылкой, может превратиться в нетривиальную задачу.

Использование инструментальных средств проверки позволяет значительно повысить эффективность труда. Достаточно установить расширение браузера, способное выводить информацию о результатах выполнения HTTP-запросов и выделять запросы, завершившиеся неудачей.

Наличие средств самодиагностики позволяет сократить время устранения ошибок на стадии поддержки приложения. В веб-ориентированное приложение может быть добавлена функция оповещения администратора о случаях запроса пользователями несуществующих страниц. Своевременный анализ таких ситуаций позволяет диагностировать и устранять ошибки на ранних стадиях их развития, а также косвенно помогает определить наличие проблем с безопасностью приложения.

Проверку ссылок следует повторять время от времени. В процессе интенсивной эксплуатации сложного

веб-ориентированного приложения часто возникают ситуации, способствующие возникновению проблем со ссылками (поскольку невозможно полностью исключить явное указание ссылок в статическом виде).

Из этого факта следует необходимость выполнять диагностику (или самодиагностику) приложения не только при вводе его в эксплуатацию, но и в процессе эксплуатации с некоторой периодичностью. Решение сопутствующих этой процедуре технических проблем значительно упрощается, если приложение реализовано с возможностью автоматической самодиагностики с настраиваемыми периодами повторения.

Особое внимание следует уделять внешним ссылкам. Диагностика работоспособности приложения осложняется тем фактом, что некоторые информационные источники могут быть расположены вне информационного пространства приложения. В таком случае система управления информационным наполнением приложения должна быть снабжена функцией проверки корректности любых внешних ссылок (как минимум – на стадии их добавления, чтобы исключить возникновение проблем, вызванных опечатками и т.п. ошибками администратора).

Особый вид ссылок. Сложнее всего автоматизировать самодиагностику некорректных ссылок в шаблонах и JavaScript. Однако такие ссылки, как правило, не подвержены интенсивному изменению, а потому тщательная проверка автоматизированным средством диагностики перед сдачей приложения в эксплуатацию оказывается достаточной в большинстве случаев.

ЛИТЕРАТУРА

- [1] Боровский, А. Введение в искусство модульного тестирования на PHP / А. Боровский // Web-разработка, Хабрахабр [Электронный ресурс]. 2009. - Режим доступа: <http://habrahabr.ru/blogs/webdev/59579/>. - Дата доступа: 24.09.2009.
- [2] Савин, Р. Тестирование дот ком / Р. Савин // СПб.: "Дело". С116, 2007 г. 312 с.
- [3] Блэк, Р. Ключевые процессы тестирования / Р.Блэк // М.: "Лори". Москва, 2006 г. 544 с.