

Тестирование WEB-Приложений

Что называется web-приложением?



Веб-приложение — клиент-серверное приложение, в котором клиентом выступает браузер, а сервером — веб-сервер. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются кроссплатформенными сервисами.

Тестирование WEB-Приложений

Какими бывают Web - приложения?

1. «Простые» сайты и веб-приложения:
 - Информационные сайты
 - Электронные магазины
 - Простые веб-приложения



Тестирование WEB-Приложений

2. Комплексные приложения и порталы

- Решения с расширенной функциональностью
- Горизонтальные порталы и социальные сети
 - Решения по потоковой передаче данных
 - Интернет-аукционы и торговые площадки



Тестирование WEB-Приложений

3. Веб-продукты повышенной сложности и облачные решения

- Инновационные продукты
 - SaaS-решения
- Поисковые системы
- Трейдинговые системы
- Системы онлайн-платежей



Клиент-серверная архитектура

2-х уровневая клиент-серверная архитектура



Клиент-серверная архитектура

2-х уровневая клиент-серверная архитектура



Тестирование WEB-Приложений

С чего состоит тестирование WEB-приложений?



Тестирование WEB-Приложений

1. Подготовительные работы

тестировщик изучает полученную документацию (анализирует функционал по тех. заданию, изучает конечные макеты сайта и составляет план теста для дальнейшего тестирования)

2. Функциональное тестирование – наиболее продолжительный этап проверки ресурса. Суть этого процесса заключается в проверке всего описанного функционала:

Проверки работы всех обязательных функций сайта;

Тестирования работоспособности пользовательских форм на сайте (например, обратная связь, добавление комментария в блог);

Проверки работы поиска (включая релевантность результатов);

Проверки гиперссылок, поиск нерабочих ссылок;

Проверки подгрузки файлов на сервер;

Проверки работоспособности счётчиков, установленных на страницах сайта;

Просмотр на соответствие содержимого страниц сайта исходному контенту, предоставленному заказчиком.

3. Тестирования Верстки – при проверке верстки первым делом тестировщик проверяет расположения элементов, соответствие их позиций предоставленным макетам, а так же проверяет оптимизацию изображений и графики. Далее осуществляется проверка валидности кода.

В процессе вёрстки важно соблюдать корректную иерархию объектов, и важно удостовериться в её валидности по факту завершения работ. Браузеры, несмотря на явно неверный код, в любом случае постараются отобразить веб-страницу. Но поскольку не существует единого регламента о том, как же должен быть показан «кривой» документ, каждый браузер пытается сделать это по-своему. А это в свою очередь приводит к тому, что один и тот же документ может выглядеть по-разному в различных браузерах.

Исправление явных промахов и систематизация кода приводит, как правило, к стабильному результату. Завершив проверку на валидность, специалист приступает к проверке на кроссбраузерность, т.е. проверяет работоспособность сайта в различных браузерах, а так же при различных параметрах настройки экрана.



Зачем проверять сайт на кроссбраузерность? На сегодняшний день существует ряд наиболее популярных веб-браузеров, таких как Google Chrome, Safari, Mozilla Firefox, Internet Explorer и Opera. Каждый из них придерживается общих рекомендаций визуализации разметки страницы, однако в то же время каждый обрабатывает код в соответствии с особенностями собственного движка. Осложняется всё тем, что достаточно часто появляются новые версии браузеров, и ресурс, который отлично смотрится, к примеру, в IE9, не обязательно будет выглядеть корректно в IE7 или IE8. Поэтому в процессе тестирования учитывается перечень браузеров, поддержка которых оговаривалась с заказчиком на ранних этапах обсуждения проекта. Этап проверки сайта на кроссбраузерность при различных разрешениях достаточно долгий по времени, но результат того стоит – с вашим сайтом сможет ознакомиться любой представитель целевой аудитории.

Usability тестирование – проводится для оценки удобства продукта в использовании, основанный на привлечении пользователей в качестве тестировщиков и анализ полученных результатов.

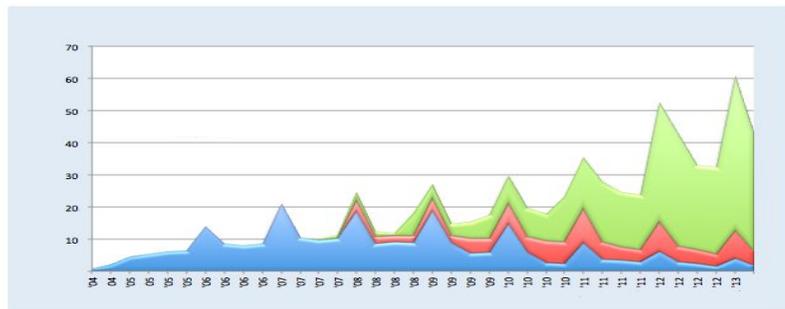
Несмотря на тот факт, что проработка удобства использования ресурса осуществляется в процессе составления технического задания, разработки макетов, бывают ситуации, когда полученный результат не является оптимальным. Хотя такое и происходит достаточно редко, оптимальное решение в данном случае – внести изменения в реализованный продукт.

Тестирование проводится с участием нескольких человек из целевой аудитории, так называемых респондентов. Для проведения тестирования достаточно 4-6 человек. Существует правило 80/20, которое гласит, что 20% пользователей дают 80% результата. Поэтому такое количество респондентов максимально эффективно с точки зрения экономии времени и затрат.

Тестирование безопасности – На данной стадии тестирования специалист проверяет – нет ли у пользователей доступа к служебным/закрытым страницам а так же проводит проверку защиты всех критически важных страниц (например, раздела администрирования сайта) от внешнего воздействия.



Тестирование производительности сайта — проводится с целью определения быстродействия сайта или его части под определённой нагрузкой. Тестирование производительности включает в себя такие виды тестирования:



Нагрузочное тестирование — простейшая форма тестирования производительности. Нагрузочное тестирование обычно проводится для того, чтобы оценить поведение сайта (или приложения) под заданной ожидаемой нагрузкой. Этой нагрузкой может быть, например, ожидаемое количество одновременно работающих пользователей на сайте, совершающих заданное число транзакций за интервал времени. Такой тип тестирования обычно позволяет получить время отклика всех самых важных бизнес-функций.

Тестирование быстродействия — проверка скорости загрузки сайта для определения скорости отработки скриптов, загрузки изображений и контента. Этот тест проводится с целью оптимизации процесса загрузки сайта, а так же определения оптимальности настроек сервера.

Технологические отличия.

Классическое приложение работает с использованием одной или семейства родственных технологий.

Web-приложение работает с использованием принципиально различных технологий.

Структурные отличия.

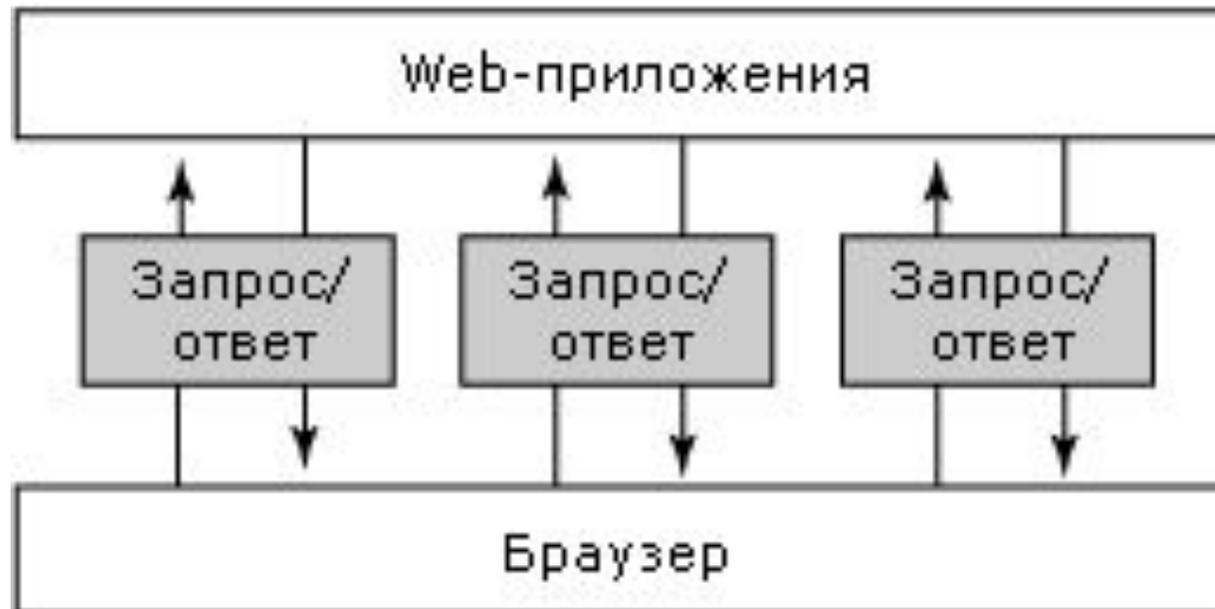
Классическое приложение “монолитное”. Состоит из одного или небольшого количества модулей. Не использует серверы БД, web-серверы и т.д.

Web-приложение – “многокомпонентное”. Состоит из большого числа модулей. Обязательно использует серверы БД, web-серверы, серверы приложений.

Отличия режимов работы.

Классическое приложение работает *в режиме реального времени*, т.е. известно о действиях пользователя сразу же, как только оно выполнено.

Web-приложение работает в режиме “запрос-ответ”, т.е. известно о некотором наборе действий только после запроса на сервер.



Отличия формирования интерфейса.

Классическое приложение использует для формирования интерфейса пользователя относительно устойчивые и стандартизированные технологии.

Web-приложение использует для формирования пользовательского интерфейса стремительно развивающиеся технологии, множество которых конкурирует между собой.

Отличия работы с сетью.

Классическое приложение практически не использует сетевые каналы передачи данных.

Web-приложение активно использует сетевые каналы передачи данных.

Отличия запуска и остановки.

Классическое приложение запускается и останавливается редко.

Web-приложение запускается и останавливается по факту поступления каждого запроса, т.е. очень часто.

Разница в количестве пользователей.

Классическое приложение: количество пользователей, одновременно использующих приложение, подвержено контролю, ограничено и легко прогнозируемо.

Web-приложение: количество пользователей, одновременно использующих приложение, сложнопрогнозируемо и может скачкообразно меняться в широких диапазонах.

Особенности сбоев и отказов.

Классическое приложение: выход из строя тех или иных компонентов сразу становится очевидным.

Web-приложение: выход из строя некоторых компонентов оказывает непредсказуемое влияние на работоспособность приложения в целом.

Отличия в установке.

Классическое приложение — процесс установки стандартизирован и максимально ориентирован на широкую аудиторию пользователей. Не требует специфических знаний. Добавление компонентов приложения выполняется стандартным способом с использованием одного и того же установщика.

Web-приложение — процесс установки часто недоступен конечному пользователю. Установка требует специфических знаний. Процесс изменения компонент приложения не предусматривается или требует квалификации пользователей. установщик отсутствует.

Отличия в деинсталляции.

Классическое приложение: процесс деинсталляции стандартизирован и выполняется автоматически или полуавтоматически.

Web-приложение: процесс деинсталляции требует специфических знаний для вмешательства администратора и часто сопряжен с изменением кода среды функционирования приложения, БД, настройки системного ОС.

Особенности среды функционирования.

Классическое приложение: среда функционирования стандартизирована и не сильно влияет на функционирование приложения.

Web-приложение: среда функционирования очень разнообразна и может оказать серьезное влияние на работоспособность и серверной, и клиентской части.