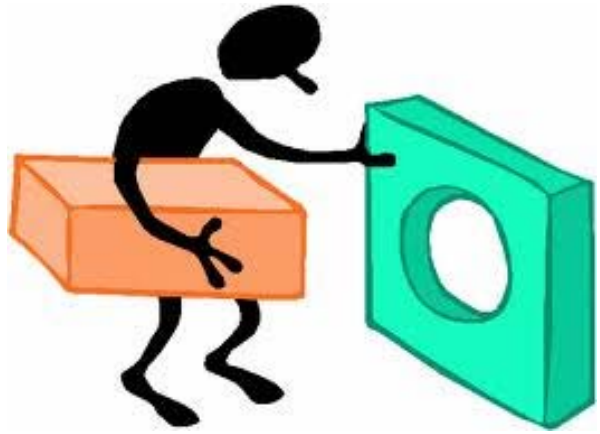
Тема 5
«Тестирование совместимости»

Вступление

Тестирование совместимости (compatibility testing) – проверка того, *как приложение взаимодействует с другими приложениями и операционной системой*.

В случае веб-ориентированных приложений особое внимание уделяется совместимости с различными браузерами.



Особенности взаимодействия с различным клиентским ПО

Определения

Клиентской частью веб-ориентированного приложения может выступать любое ПО, способное отправить http-запрос, и обработать полученный ответ.

Сложности начинаются со слова «обработать».

Во-первых, клиентское ПО может предоставлять результат обработки человеку, а может использовать его самостоятельно.

Во-вторых, клиентского ПО много. ОЧЕНЬ много.

Кроссбраузерное тестирование

Определение

Кроссбраузерное тестирование – проверка способности веб-ориентированного приложения идентично работать и отображаться во всех заявленных поддерживаемых браузерах (обычно учитываются лишь *наиболее распространённые*).

Под идентичностью понимается отсутствие нарушений вёрстки и функционирования JS, Flash и т.п.

Следует обратить внимание на тот факт, что кроссбраузерная совместимость не обязательно означает «попиксельное соответствие» внешнего вида приложения в разных браузерах.

Как обеспечить кроссбраузерность

Наиболее распространённый (но не самый лучший) способ – написание т.н. «хаков», т.е. специального CSS/JS-кода, который будет выполняться только в определённых версиях определённых браузеров.

Иногда генерацию соответствующего кода перекладывают на серверную часть приложения.

Как обеспечить кроссбраузерность

Поскольку браузеров ОЧЕНЬ много, писать свою логику поведения и отображения приложения для каждого из них – нерационально.

Поэтому рекомендуется использовать такие элементы вёрстки, которые одинаково хорошо совместимы с большинством версий наиболее распространённых браузеров.

Как обеспечить… : три подхода

Как уже было говорено – браузеров ОЧЕНЬ много, потому обеспечение кроссбраузерности можно реализовывать так:

Подход-1 (правильный): чётко планировать соответствующие действия команды разработки и тестирования, проводить полномасштабные тесты.

Подход-2 (ускоренный для опытных): держать в голове самые распространённые проблемы самых распространённых браузеров; писать так, чтобы избежать этих проблем.

Как обеспечить… : три подхода

Подход-3 (ускоренный (?!) для начинающих): писать так, чтобы приложение было совместимо с 2-3 самыми распространёнными браузерами и их версиями.

По мере выявления проблем с другими версиями и браузерами – «латать дыры», т.е. писать «хаки» CSS/JS для этих версий.

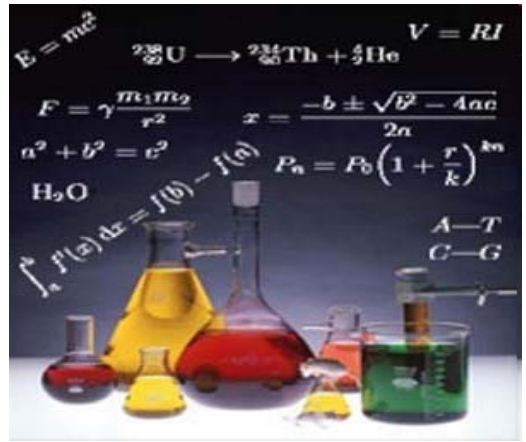
Этот подход в масштабных проектах может оказаться дороже первого… в разы.

- Подскажите, как с мака удалить Windows Vista.
- С наслаждением!

(C) bash.org.ru

Как проверить кроссбраузерность

Научный подход: поставить мощный сервер, на котором в виртуальных машинах под разными операционными системами выполняются разные браузеры. Подключаться удалённо к этим виртуальным машинам и открывать сайт.

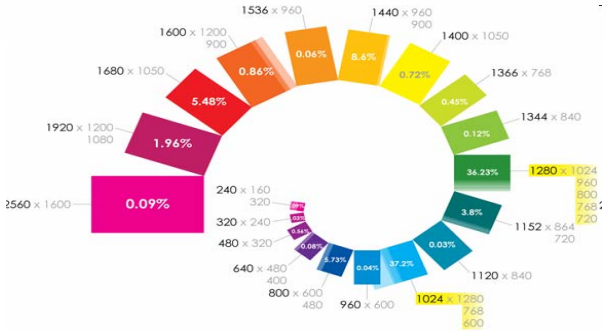


Упрощённый подход: просто подготовить много образов виртуальных машин, запускать их на своём компьютере по мере надобности.

Самый простой подход: использовать готовые сервисы.

Как проверить кроссбраузерность

Т.о. в вашем приложении при тестировании кроссбраузерной совместимости нужно проверить:



- как приложение отображается и работает в различных браузерах (семейство Mozilla, Internet Explorer, Opera, Safari, Chrome, мобильные браузеры);
- как приложение отображается и работает при различных разрешениях экрана (обычно 1024×768, 1280×800 и т.д.)
- как приложение отображается и работает в различных операционных системах (Mac OS, Linux, Win).

Готовые инструменты

О валидации кода мы уже говорили в теме 3 «*Тестирование по методу белого ящика*». Добавим только, что валидация не гарантирует кроссбраузерности, но часто помогает «стать на нужный путь».

А вот непосредственно в тестировании кроссбраузерности нам помогут он-лайн сервисы и пара оффлайновых утилит.

Он-лайн сервисы

IE NetRenderer – показывает (генерирует изображения), как сайт будет выглядеть под различными версиями MSIE:

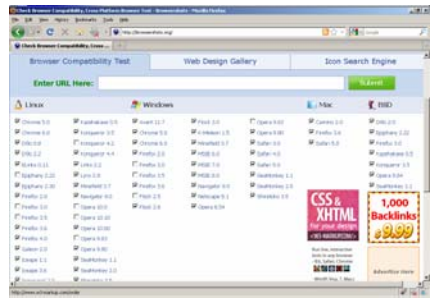
http://ipinfo.info/netrenderer/

Browsershots.org – пожалуй, самый известный сервис кроссбраузерного тестирования.

Поддерживает десятки версий самых разнообразных браузеров под самыми разнообразными операционными системами.

В очень удобном виде генерирует изображения:

http://browsershots.org



Он-лайн сервисы – список

В дополнение – список подобных сервисов (по материалам http://womtec.ru/2010/06/online-tools-test-cross-browser-compatibility/):

Browser Sandbox (**http://spoon.net/Browsers/**) – позволяет открывать станицы в разных браузерах, таких как IE, Firefox, Safari, Chrome и Opera непосредственно из своего браузера.

Browsrcamp (**http://www.browsrcamp.com/**) – позволяет проверить совместимость приложения с браузерами в Mac OS X.

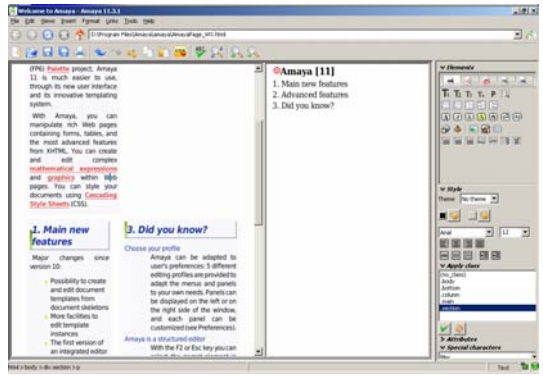
Adobe BrowserLab (**https://browserlab.adobe.com/en-us/index.html**) – позволяет проверять отображение сайтов в разных браузерах и на разных ОС.

Browsera (**http://www.browsera.com/**) – платный онлайн сервис предназначенный для тестирования сайта в целом, а не только для проверки кроссбраузерности сайта.

Litmusapp (**http://litmusapp.com/**) – платный сервис с большим количеством разнообразных функций, как мелких и довольно специфичных, так и крупных и очень полезных (в т.ч. возможность тестирования сайта на мобильных устройствах с Windows Mobile, Symbian или iPhone).

Офф-лайн инструменты

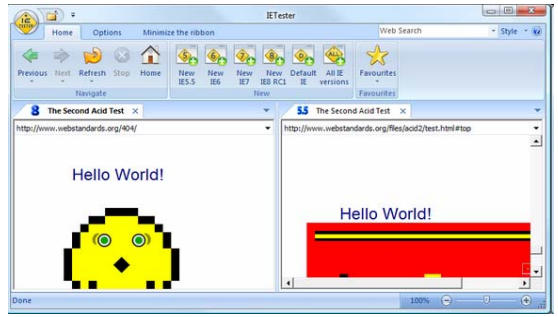
Amaya – браузер от W3C. Позволяет хорошо изучить клиентскую часть приложения на предмет того, что в ней есть, и как это работает.



Бесплатно загрузить последнюю версию можно здесь:

http://www.w3.org/Amaya/

IETester – бесплатный веб-браузер, который содержит в себе несколько версий Internet Explorer: IE9, IE8, IE7, IE 6 и IE5.5 для Windows 7, Vista и XP.

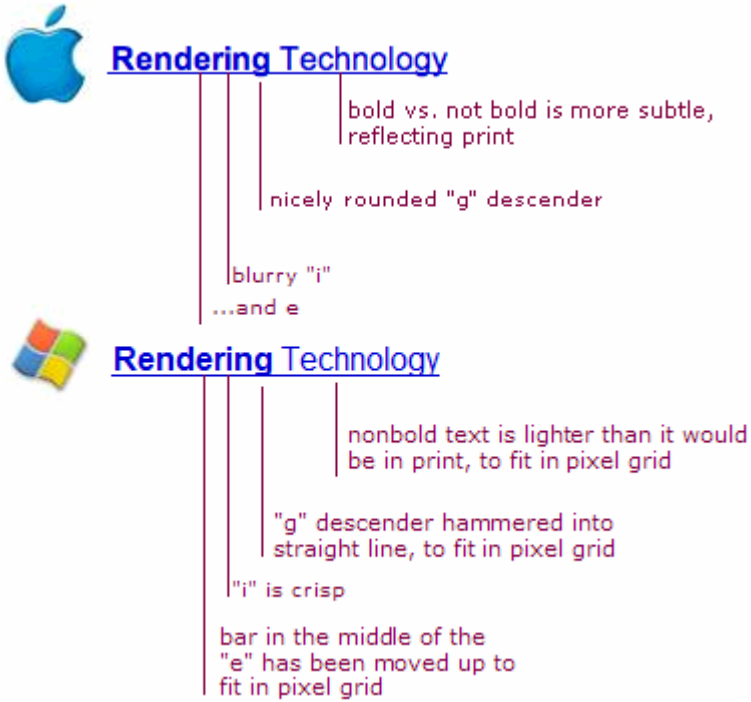


http://www.my-debugbar.com/wiki/IETester/HomePage

О чём стоит помнить

Различные браузеры и операционные системы используют различные способы вывода шрифтов.

Размеры шрифтов тоже не одинаковы в различных системах, а некоторые шрифты могут просто отсутствовать на компьютере пользователя.



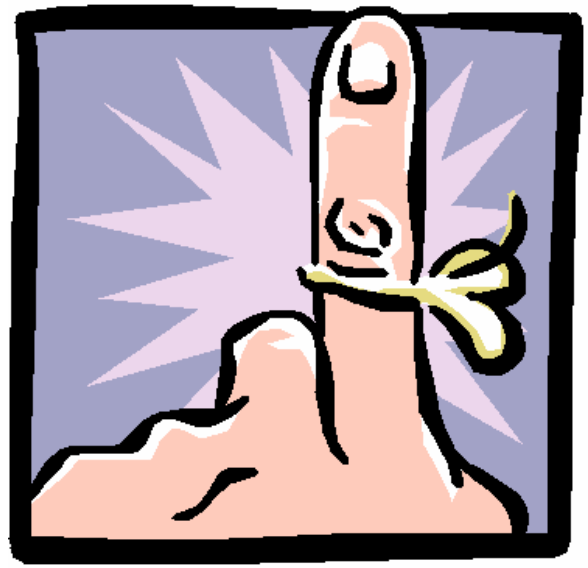
О чём стоит помнить

В различных браузерах могут быть свои особенности использования JavaScript.

В частности, для использования AJAX нужно предусматривать несколько способов получения экземпляра XMLHttpRequest:

```
function CreateXMLHTTP() {
  var xh = null;
  try {
    xh = new ActiveXObject("Msxml2.XMLHTTP"); }
    catch(e) {
       try {
         xh = new ActiveXObject("Microsoft.XMLHTTP");
          }
    catch(e) {
         xh = null;
              }
       }
  if(!xh && typeof XMLHttpRequest != "undefined") {
    xh = new XMLHttpRequest();
    }
  if (!xh) return false; return xh;
}
```

О чём стоит помнить



Помимо вёрстки стоит помнить о:

- Cookies – приложение должно быть полностью работоспособным, если они у пользователя выключены.
- JavaScript – если он выключен, приложение должно продолжать работать.

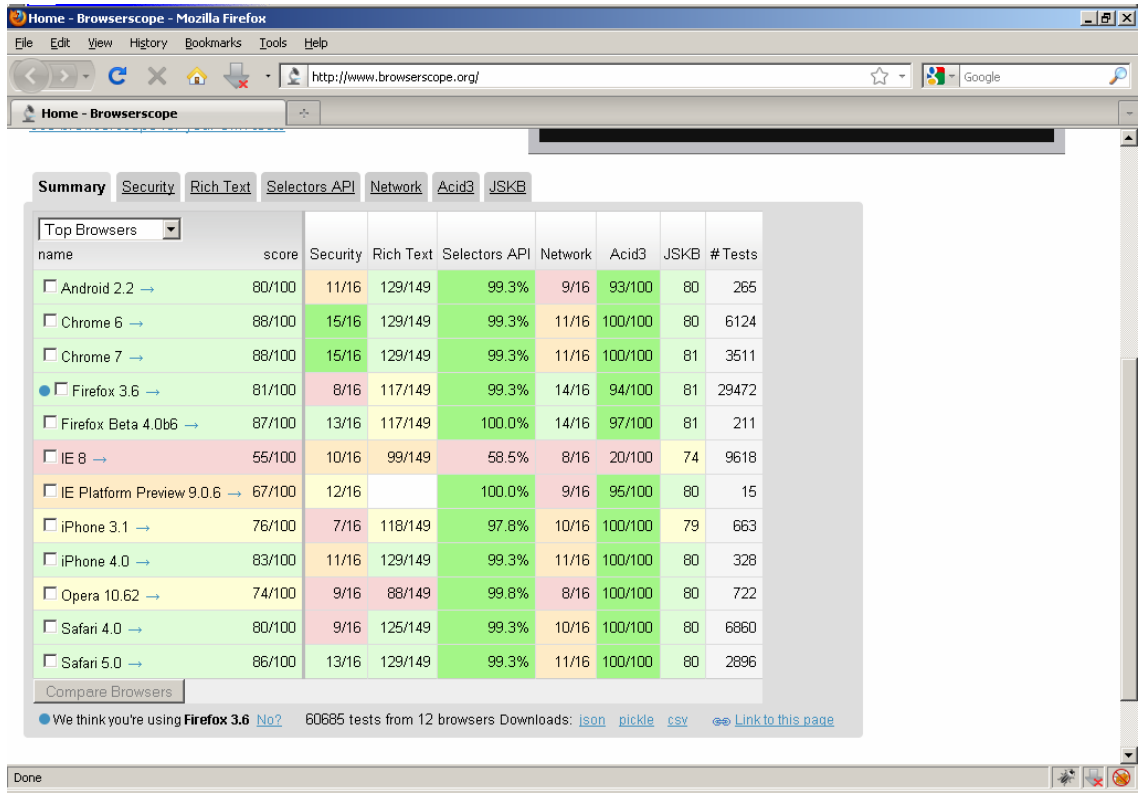
- Без крайней необходимости приложение не должно обращаться куда бы то ни было по порту, отличному от 80-го.
- Приложение должно иметь возможность работать БЕЗ установки в браузер дополнительных плагинов.

Как узнать, в чём проблемы браузера

Если про некоторую версию браузера вы знаете мало, имеет смысл её протестировать.

Для этого есть хороший сервис:

http://www.browserscope.org/



Тест для проверки изученного

1. Дайте определение тестирования совместимости.
2. Дайте определение кроссбраузерного тестирования.
3. Назовите основные способы обеспечения кроссбраузерности.
4. Каковы основные подходы к тестированию кроссбраузерности?
5. Какие готовые инструменты кроссбраузерного тестирования вы знаете?
6. Как выяснить, какие технологии поддерживает тот или иной браузер?
7. Как проверить сайт сразу в нескольких браузерах?
8. Можно ли автоматизировать кроссбраузерное тестирование?
9. На финальной стадии работы с проектом вдруг выявляется серьёзная несовместимость с неким набирающим популярность браузером – как поступить в этой ситуации?
10. Какие проблемы с совместимостью может принести «нестандартная» реализация JavaScript в том или ином клиентском ПО?

Тестирование совместимости с мобильными устройствами, эмуляторы

Проблемы с мобильными устройствами

Итак, с обычным клиентским ПО мы разобрались. Но ещё есть клиентское ПО в мобильных устройствах (смартфонах, КПК, обычных телефонах и т.п.).

Основных проблемы здесь две:

1. Таких устройств и клиентского ПО на них – много, и все они – разные.

2. Как правило, клиентское ПО на таких устройствах обладает упрощённой функциональностью по сравнению со своими «полноценными аналогами».

Решение проблем

Обе эти проблемы имеют одно очевидное решение – нужно проверить совместимость приложения:

1. Своими силами – с наиболее известными мобильными платформами.

2. Силами добровольцев – с малораспространёнными мобильными платформами.

Как проверять

Для того, чтобы проверить работоспособность приложения под той или иной платформой, нам нужно либо само мобильное устройство, либо его эмулятор.

При этом эмулятор может быть двух видов:

- эмулятор всего устройства целиком;
- эмулятор браузера (или иного веб-клиента) для некоторой платформы.

Эмуляторы устройств

Поскольку данная предметная область слишком обширна и постоянно претерпевает сильные изменения, скажем кратко: если ваша целевая аудитория активно пользуется неким устройством – ищите, скачивайте, устанавливайте и используйте его эмулятор.

Google в помощь! :)

Эмуляторы устройств

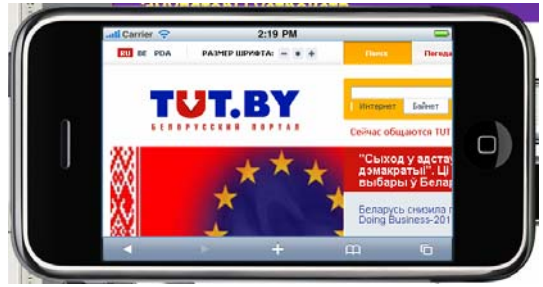
Наиболее известными «эмуляторами устройств целиком» на сегодняшний день являются:

iBBDemo (эмулятор iPhone)

http://www.puresimstudios.com/ibbdemo/

Демонстрационный ролик:

http://labs.blackbaud.com/ibbdemo/ibbdemo.html



iPhoney (эмулятор iPhone для MacOS)

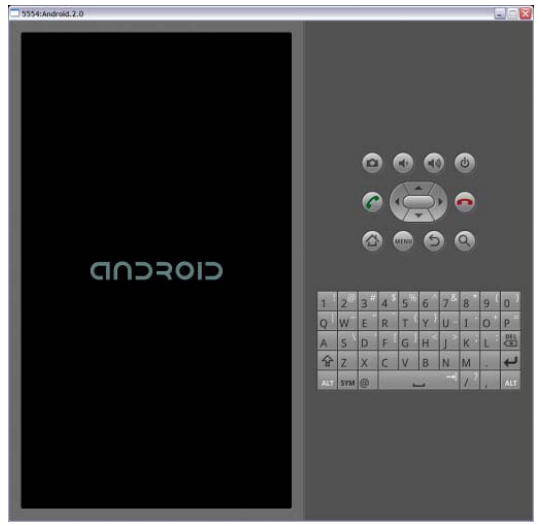
http://www.marketcircle.com/iphoney/

Возможно, полезными окажутся инструменты от самой фирмы Apple:

http://developer.apple.com/technologies/tools/

Эмуляторы устройств

Не менее интенсивно развивается и платформа Android.



Эмулятор Android:

http://developer.android.com/guide/developing/tools/emulator.html

http://phoneandroid.ru/info/android-emulator#more-589

Эмуляторы мобильных браузеров

В качестве примера рассмотрим эмулятор Opera Mobile Emulator:

http://www.opera.com/developer/tools/

Opera Mini Demo (он-лайн сервис, эмулирующий работу Opera Mini):

http://www.opera.com/mini/demo/

И, в заключение, рекомендуется почитать прекрасную статью (см. файл "Эмуляторы для тестирования и разработки мобильных веб-сайтов и приложений.pdf")

Тест для проверки изученного

1. Какие виды эмуляторов для тестирования совместимости с мобильными устройствами вы знаете?
2. Приведите несколько примеров эмуляторов каждого вида.
3. Если для какого-то устройства эмулятора нет – что делать?
4. Гарантирует ли использование при тестировании эмулятора, что на реальном устройстве всё то, что хорошо работало на эмуляторе, будет работать так же хорошо?
5. Какие тесты совместимости с мобильными устройствами вы бы проводили в первую очередь?

Факторы, вызывающие потерю совместимости, и способы их устранения

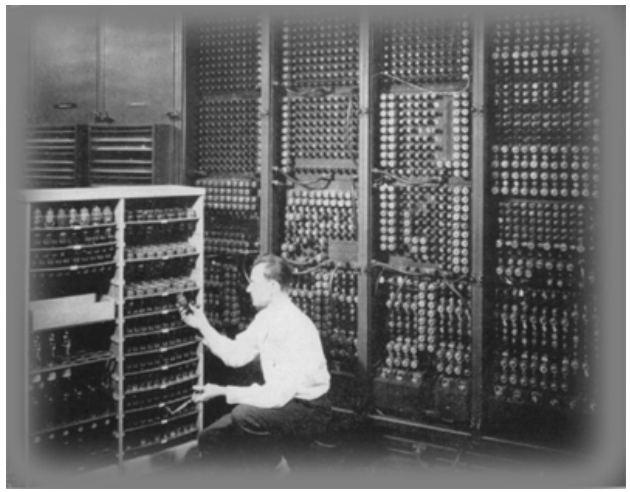
Устаревшие технологии

Проблема

Использование устаревших технологий, поддержка которых либо официально прекращена, либо осуществляется формально.

Решение

Идти в ногу со временем, периодически пересматривать старые библиотеки кода, проверяя их на совместимость с современным ПО.



Самые новые технологии

Проблема

Использование самых новых технологий, ещё не нашедших повсеместной поддержки, что приводит к потере совместимости со ВСЕМ клиентским ПО, ещё не начавшим поддерживать эту технологию.

Решение

Учитывать ситуацию на рынке клиентского ПО. Применять, по возможности, те технологии, которые хорошо зарекомендовали себя в большинстве активно используемого клиентского ПО.

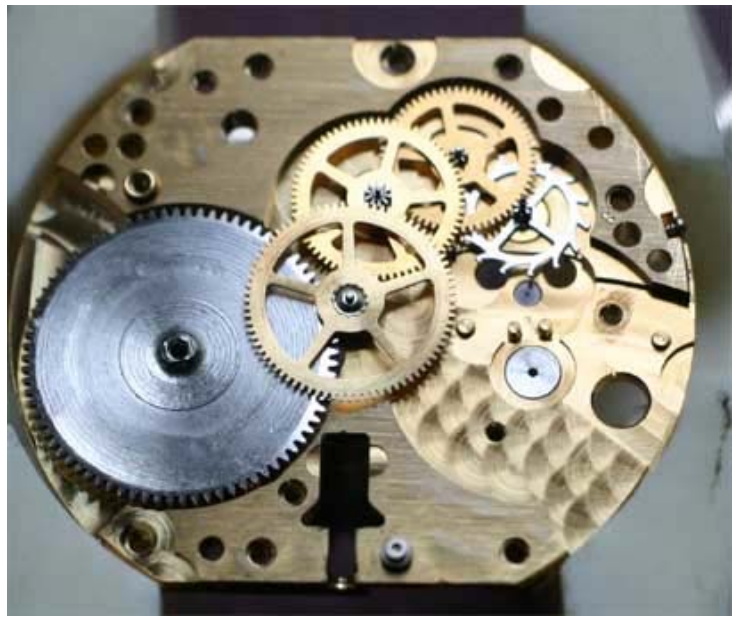
Нарушение стандартов

Проблема

Нарушение стандартов. Всё же индустрия движется ко всё большей и большей стандартизации в области широкораспространённых технологий. «Идти своим путём» и изобретать велосипед – порочная практика.

Решение

Следовать стандартам. Проверять, насколько наше приложение им соответствует.



Отсутствие учёта особенностей

Проблема

Отсутствие учёта особенностей конкретных видов и версий клиентского ПО. Несмотря на тенденцию ко всеобщей стандартизации, некоторые производители клиентского ПО по тем или иным причинам привносят в свои разработки «изюминки», которые, по факту, становятся проблемами.

Решение

Глубоко изучать наиболее распространённые виды клиентского ПО, знать их особенности и использовать эти знания при разработке собственных приложений.

Дополнительные компоненты

Проблема

Применение технологий, требующих установки дополнительных компонентов в клиентское ПО.

Если клиентское ПО или его пользователь по какой-то причине отказывается устанавливать и применять дополнительные компоненты, функционал нашего приложения, опирающийся на такие компоненты, перестанет работать.

Решение

Несмотря на то, что технологии наподобие Flash стали повсеместно распространёнными, приложение не должно вести себя так, будто они будут доступны всегда, везде и безусловно.

Должны быть продуманы альтернативные способы функционирования при недоступности «привычных дополнительных компонентов».

Нетипичная работа с сетью

Проблема

Использование нетипичного для веб-ориентированных приложений сетевого взаимодействия часто приводит к конфликтам со средствами безопасности или общей инфраструктурой сети (например, клиенту разрешены соединения только по http:80).

Решение

Отказ от использования нетипичного сетевого взаимодействия или разработка альтернативных способов выполнения задач, требующих такого нетипичного взаимодействия.

Отсутствие учёта предпочтений

Проблема

Отсутствие учёта предпочтений пользователей по настройкам того или иного клиентского ПО.

Если большинство пользователей предпочитает некий набор несовместимых с нашим приложением настроек, наше приложение не будет работать корректно у большей части его целевой аудитории.

Решение

Изучение целевой аудитории. Сбор и анализ статистики с целью выявления наиболее характерных профилей настроек клиентского ПО.

Плохой код

Проблема

Создание плохо поддерживаемого кода HTML/CSS/JS и серверной части приложения, что приводит к сложнопредсказуемым последствиям при внесении изменений.

В результате этой ситуации уже вполне качественное с точки зрения совместимости приложение может в одночасье «посыпаться».

Решение

Постоянное внимание вопросам качества процесса разработки, сопровождаемости кода, гибкости архитектуры и т.п.

Понимание того факта, что в приложении всё взаимосвязано, и невозможно гарантировать высокие показатели совместимости, если мы не уверены в качестве тех частей приложения, которые отвечают за генерацию кода для клиентской части.

Недостаток внимания тестированию

Проблема

Недостаточное внимание вопросам тестирования совместимости.

Эта проблема прямо и косвенно приводит к разрастанию всех перечисленных ранее проблем и сложностей.

Решение

Уделять вопросам совместимости столько сил и времени, сколько требуется для достижения установленных показателей качества.

Факторы: все в одном списке

Итак, что может ухудшить совместимость веб-ориентированного приложения с клиентским ПО:

- использование устаревших технологий;
- использование самых новых технологий, ещё не нашедших повсеместной поддержки;
- нарушение стандартов;
- отсутствие учёта особенностей конкретных видов и версий клиентского ПО;
- применение технологий, требующих установки дополнительных компонентов в клиентское ПО;
- использование нетипичного для веб-ориентированных приложений сетевого взаимодействия;
- отсутствие учёта предпочтений пользователей по настройкам того или иного клиентского ПО;
- создание плохо поддерживаемого кода HTML/CSS/JS и серверной части, что приводит к сложнопредсказуемым последствиям при внесении изменений;
- недостаточное внимание вопросам тестирования совместимости.

Тест для проверки изученного

1. Назовите как можно больше факторов, вызывающих потерю совместимости веб-ориентированного приложения с клиентским ПО.
2. Как можно проверить сетевую активность веб-ориентированного приложения?
3. С помощью чего можно проверить соответствие кода клиентской части веб-ориентированного приложения стандартам?
4. Если некоторое распространённое клиентское ПО обладает особенностями, идущими вразрез со стандартами, – что делать в такой ситуации?
5. В какой момент разработки веб-ориентированного приложения тестирование совместимости окажется наиболее эффективным?
6. Какие тесты совместимости имеет смысл автоматизировать?
7. Назовите несколько инструментальных средств, применяемых при тестировании совместимости.
8. Как показатели совместимости влияют на показатели юзабилити?
9. Приведите несколько примеров того, как проблемы с совместимостью приводят к полной неработоспособности приложения.
10. Как проблемы с совместимостью могут повлиять на показатели безопасности приложения?