

АЛЕКСАНДР КЛИМОВ

# JavaScript

2-е издание

НА ПРИМЕРАХ

ОБЩИЕ РЕКОМЕНДАЦИИ  
ПРИ СОЗДАНИИ СЦЕНАРИЯ

РАБОТА С ОБЪЕКТАМИ И ЭЛЕМЕНТАМИ  
ДОКУМЕНТА, ИЗОБРАЖЕНИЯМИ,  
ДАТОЙ И ВРЕМЕНЕМ, СТРОКАМИ

СПЕЦЭФФЕКТЫ ДЛЯ ИЗОБРАЖЕНИЙ  
И ТЕКСТА

ШУТОЧНЫЕ ПРИМЕРЫ И ИГРЫ  
НА JavaScript

СОЗДАНИЕ ИНТЕРАКТИВНЫХ  
Web-СТРАНИЦ

РАСШИРЕНИЯ ДЛЯ БРАУЗЕРОВ

ОБЛАСТИ ПРИМЕНЕНИЯ JavaScript

СОВЕТЫ И ХИТРОСТИ  
ПРИ СОЗДАНИИ СЦЕНАРИЕВ



СКАЧАЙ ФАЙЛЫ  
ПРИМЕРОВ  
[www.bhv.ru](http://www.bhv.ru)

УДК 681.3.06  
ББК 32.973.26-018.2  
К49

**Климов А. П.**

К49 JavaScript на примерах. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 336 с.: ил.

ISBN 978-5-9775-0361-7

На примерах рассмотрены методы разработки сценариев на языке JavaScript. Представлены нестандартные приемы работы с объектами и изображениями, примеры работы с датами и системными настройками, создание спецэффектов и др. Уделено внимание разработке шуточных программ и игр. Показано создание интерактивных Web-страниц, получение сведений о системе и браузере, создание расширения для браузеров. Приведены практические советы по работе с JavaScript. Все примеры написаны с учетом особенностей двух популярных браузеров: Internet Explorer и Mozilla Firefox. Во втором издании появились новые и переработаны "старые" примеры с учетом появления новых ОС и браузеров.

*Для веб-разработчиков*

УДК 681.3.06  
ББК 32.973.26-018.2

#### **Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Анна Кузьмина</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн серии	<i>Игоря Цырульников</i>
Оформление обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.09.08.

Формат 70×100<sup>1</sup>/<sub>16</sub>. Печать офсетная. Усл. печ. л. 27,09.

Тираж 2500 экз. Заказ №

"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Отпечатано с готовых диапозитивов  
в ГУП "Типография "Наука"  
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0361-7

© Климов А. П., 2008

© Оформление, издательство "БХВ-Петербург", 2008

# Оглавление

<b>ВВЕДЕНИЕ.....</b>	<b>1</b>
Обращение к читателю.....	1
Для кого эта книга .....	1
О браузерах.....	1
Как пользоваться примерами.....	2
Благодарности .....	2
 <b>ГЛАВА 1. ПЕРВОЕ ЗНАКОМСТВО С JAVASCRIPT.....</b>	<b>3</b>
1.1. Первые приготовления .....	3
1.2. Первый сценарий .....	4
1.3. Разбор полетов .....	5
1.4. Скрытие сценария .....	7
1.5. Комментарии в JavaScript .....	7
1.6. Проблемы .....	8
1.7. Сценарий в действии .....	10
 <b>ГЛАВА 2. ИНФОРМАЦИЯ О СИСТЕМЕ.....</b>	<b>11</b>
2.1. Война браузеров.....	11
2.2. Информация о системе и браузере.....	11
2.3. Определение браузера .....	15
2.4. Свойства экрана .....	18
2.5. Размеры документа .....	20
2.6. Информация об операционной системе .....	21
2.7. Свойство <i>userAgent</i> .....	22
2.7.1. Определение версии Mozilla Firefox .....	26
2.7.2. Определение версии Opera.....	27
2.7.3. Дополнительные маркеры.....	28
2.8. Определение версии JScript .....	29
2.9. Проверка на наличие расширений .....	30
2.10. Добавить в <i>Избранное</i> .....	31

2.11. Вывод диалоговых окон <i>Выбор языка</i> и <i>Упорядочить Избранное</i> .....	32
2.12. Установка домашней Web-страницы.....	33
<b>ГЛАВА 3. РАБОТА С ОБЪЕКТАМИ И ЭЛЕМЕНТАМИ ДОКУМЕНТА.....</b>	<b>35</b>
3.1. Немного теории.....	35
3.2. Создание нового окна браузера и загрузка в него существующей Web-страницы .....	36
3.3. Открытие окна с заданными параметрами.....	37
3.4. Создание нового окна на лету .....	38
3.5. Строка состояния .....	40
3.5.1. Информация о ссылке .....	41
3.5.2. Борьба с реферофобией.....	41
3.5.3. Смена сообщений .....	43
3.6. Заголовок .....	44
3.7. Переключатели.....	45
3.8. Текстовое поле .....	47
3.8.1. Установка фокуса при загрузке документа .....	47
3.8.2. Изменение внешнего вида текстового поля.....	47
3.8.3. Выделение текста.....	48
3.8.4. Автоматический переход на другое текстовое поле .....	48
3.8.5. Подсчет оставшихся символов .....	49
3.9. Выпадающий список .....	51
3.9.1. Навигация по Web-сайту .....	51
3.9.2. Выбор с подтверждением.....	52
3.9.3. Связывание с массивом изображений .....	53
3.9.4. Динамическое изменение элементов при использовании двух списков.....	56
3.10. Таблицы .....	58
3.11. Полосы прокрутки .....	61
3.12. Ссылки .....	62
3.12.1. Число ссылок на Web-странице.....	62
3.12.2. Запрет открытия ссылки в новом окне .....	63
3.13. Переопределение стандартного поведения ссылки.....	65
3.14. Создание удобного интерфейса.....	65
3.15. Изменение фона Web-страницы.....	69
3.15.1. Изменение фона случайным образом .....	70
3.16. Работа с буфером обмена.....	72
3.17. Обработка нажатий клавиш.....	75

<b>ГЛАВА 4. РАБОТА С ИЗОБРАЖЕНИЯМИ .....</b>	<b>81</b>
4.1. Проверка на возможность загрузки изображений .....	81
4.2. Массив изображений .....	82
4.3. Создание эффекта прозрачности .....	84
4.4. Создание слайд-шоу .....	85
4.5. Флип-флоп .....	91
4.6. Отключение возможности вызова контекстного меню правой кнопкой мыши .....	94
4.7. Плавающая картинка .....	95
 <b>ГЛАВА 5. РАБОТА С ДАТОЙ И ВРЕМЕНЕМ .....</b>	 <b>99</b>
5.1. Создание временных задержек .....	99
5.2. Объект <i>Date</i> .....	99
5.3. Проблема 2000 года решена .....	101
5.4. Часы в строке состояния .....	101
5.5. Дата последнего изменения документа .....	103
5.6. Приветствие .....	104
5.7. Сколько дней осталось до праздника? .....	105
 <b>ГЛАВА 6. РАБОТА СО СТРОКАМИ .....</b>	 <b>109</b>
6.1. Эффект печатной машинки .....	109
6.2. Бегущая строка .....	112
6.3. Эффект волны .....	115
6.4. Эффект морской волны .....	117
6.5. Эластичный текст .....	119
6.6. Резиновый текст .....	120
6.7. Мигающий текст .....	121
6.8. Радужный текст .....	123
 <b>ГЛАВА 7. РАЗЛИЧНЫЕ СПЕЦЭФФЕКТЫ .....</b>	 <b>127</b>
7.1. Фильтры преобразования изображений .....	127
7.1.1. Постепенное растворение картинки .....	127
7.1.2. Отражение в воде .....	128
7.2. Слайд-шоу .....	130
7.3. Прокрутка фона Web-страницы .....	133
7.4. Надпись, следующая за курсором мыши .....	135
7.5. Падающий мячик .....	140
7.6. Отражение от стенок .....	143

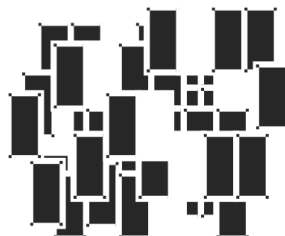
7.7. Движущиеся объекты.....	146
7.7.1. Движение по прямой .....	146
7.7.2. Движение по окружности.....	149
7.7.3. Движение по спирали .....	152
7.7.4. Движение по синусоиде .....	153
7.7.5. Движение по циклоиде.....	155
7.8. Увеличительное стекло .....	156
7.9. Мультфильм в текстовом поле .....	159
7.10. Эффект Матрицы .....	161
7.11. Падающий снег .....	166
<b>ГЛАВА 8. ШУТОЧНЫЕ ПРИМЕРЫ .....</b>	<b>171</b>
8.1. Поймай меня.....	171
8.2. Угадыватель мыслей .....	173
8.3. Программа Глаза-шпионы .....	177
8.4. Назад в будущее.....	180
<b>ГЛАВА 9. СОЗДАНИЕ ИНТЕРАКТИВНЫХ WEB-СТРАНИЦ .....</b>	<b>183</b>
9.1. Использование персонажей .....	183
9.2. Добавление команд в контекстное меню .....	186
9.3. Интерактивное поведение персонажа.....	190
<b>ГЛАВА 10. ИГРЫ НА JAVASCRIPT .....</b>	<b>195</b>
10.1. Простейшая игра.....	195
10.2. Крестики-нолики.....	198
10.3. Пятнашки.....	209
10.4. Прыгающие шарики .....	215
10.5. Найди пару.....	219
10.5.1. Правила игры.....	219
10.5.2. Создание игры .....	220
10.5.3. Создание игрового поля .....	220
10.5.4. Сценарий игры .....	221
10.5.5. Запуск новой игры .....	222
10.5.6. Начало игры.....	226
<b>ГЛАВА 11. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ.....</b>	<b>233</b>
11.1. Защищаемся от нежелательной почты .....	233
11.2. Указатели мыши .....	235

11.3. Вращающийся курсор .....	236
11.4. Закладурки или bookmarklets по-русски .....	238
11.4.1. Что такое закладурки? .....	238
11.4.2. Как пользоваться закладурками? .....	239
11.4.3. Что важно помнить? .....	239
11.4.4. Как создавать свои закладурки? .....	240
11.4.5. Кстати, а зачем нужен <i>void</i> ? .....	241
11.4.6. Фреймы: тысяча и одна проблема .....	241
11.4.7. Война браузеров .....	243
11.4.8. Сколько можно? Столько, сколько нужно .....	244
11.5. Примеры закладурок .....	245
11.5.1. Текущее время .....	245
11.5.2. Удаление фоновой картинки .....	245
11.5.3. Изменение цвета текста на Web-странице .....	246
11.5.4. Спрятать все картинки .....	246
11.6. Интернет-закладурки .....	246
11.6.1. What's — на чем работает сайт .....	247
11.6.2. Сокращения .....	247
11.6.3. Перевод .....	247
11.7. Расширения для Internet Explorer .....	248
11.7.1. Создание прямоугольников с закругленными углами .....	252
11.8. Расширения для Mozilla Firefox .....	259
11.9. Поделись улыбкою своей .....	261
11.9.1. Плагин для WordPress .....	265
11.9.2. Экспорт .....	265
11.10. Виртуальная клавиатура .....	265
11.11. Вокруг света за 80 секунд .....	266
<b>ГЛАВА 12. ДРУГИЕ ОБЛАСТИ ПРИМЕНЕНИЯ JAVASCRIPT .....</b>	<b>271</b>
12.1. Знакомство с технологией HTML Applications .....	271
12.2. Объект <i>Shell</i> .....	273
12.3. Клавиатурный тренажер .....	277
12.4. Использование сценариев в справочной системе .....	278
12.4.1. Ссылка на внешний файл .....	278
12.5. Window Script Host .....	280
12.5.1. Создание первых сценариев .....	281
12.5.2. Работа с файлами .....	282
12.5.3. Запуск программ .....	285
12.5.4. Просмотр и редактирование файлов .....	287

12.5.5. Работа с сетевым окружением .....	289
12.5.6. Сетевые принтеры .....	292
12.5.7. Работа с реестром .....	292
12.6. WMI .....	302
12.6.1. Кодеки .....	302
12.6.2. Просмотр установленных обновлений .....	303
12.6.3. Список установленных программ .....	304
12.7. Silverlight .....	306
12.8. Гаджеты для боковой панели Windows Vista .....	310
<b>ГЛАВА 13. СОВЕТЫ И ХИТРОСТИ .....</b>	<b>311</b>
13.1. Запуск сценария из адресной строки .....	311
13.1.1. Хаос на любой странице .....	311
13.2. Gmail — клиент для ссылки <i>mailto</i> .....	312
13.3. Запрет контекстного меню .....	313
13.4. Проверка на деление .....	313
13.5. Консоль JavaScript в Mozilla Firefox .....	313
13.6. Функция <i>parseInt</i> .....	314
13.7. Преобразование значений в строковый вид .....	314
13.8. Использование прототипов .....	315
13.9. Получение всех свойств объекта .....	315
13.10. Минуя все предупреждения .....	316
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>317</b>
<b>ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ .....</b>	<b>319</b>



## ГЛАВА 2



# Информация о системе

## 2.1. Война браузеров

Вероятно, вам приходилось слышать о войне браузеров. Сначала было противостояние Netscape Navigator и Internet Explorer. Затем в борьбу вступил браузер норвежских разработчиков Opera. В настоящее время идет активная вербовка в ряды сторонников модного Mozilla Firefox. Я упомянул только популярные программы для серфинга по просторам Интернета, хотя существуют еще несколько проектов независимых разработчиков.

Появление каждого нового популярного браузера приносит головную боль владельцам Web-сайта. Существует очень большая проблема совместимости, несмотря на большие усилия сообщества выработать некие стандарты для отображения Web-страниц. В результате подобной несогласованности некоторые Web-страницы по-разному отображаются в различных браузерах.

Некоторые Web-программисты, отчаявшись угодить всем посетителям Web-сайта, просто выводят надпись: "Данная страничка предназначена для просмотра только в Internet Explorer". Естественно, фанаты других браузеров не остались в долгу и закрывают доступ для владельцев детища Microsoft. О том, как можно узнать информацию об установленном браузере и других компонентах системы, чтобы использовать ее в своих целях, и будет рассказано в этой главе.

## 2.2. Информация о системе и браузере

Всю необходимую информацию о запущенном браузере и системе у пользователя можно узнать при помощи объекта `navigator`. Каждый браузер имеет несколько общих методов и свойств данного объекта, а также несколько своих,

только ему присущих свойств. Я попытался составить небольшой перечень совместимости трех браузеров: Internet Explorer, Mozilla Firefox и Opera. Сначала перечислим общие для всех свойства и методы объекта `navigator`:

- ☐ `appCodeName` — кодовое имя браузера. Обычно используется Mozilla;
- ☐ `appName` — официальное имя браузера (Internet Explorer, Netscape, Opera);
- ☐ `appVersion` — версия браузера;
- ☐ `platform` — платформа, на которой работает браузер (обычно Win32);
- ☐ `cookieEnabled` — доступность сохранения cookie<sup>1</sup>;
- ☐ `javaEnabled` — доступность на запуск сценариев JavaScript;
- ☐ `userAgent` — специальная строка для служебных целей.

Давайте на основе этой информации напишем сценарий, который будет работать в трех рассмотренных браузерах. Вставляем в теле документа следующий код (листинг 2.1).

#### Листинг 2.1. Получение информации о браузере и системе

```
<h1>Информация о браузере и системе</h1>
<script type = "text/javascript">
  var code = navigator.appCodeName;
  var name = navigator.appName;
  var vers = navigator.appVersion;
  var platform = navigator.platform;
  var cook = navigator.cookieEnabled;
  var je = navigator.javaEnabled();
  var ua = navigator.userAgent;
  document.write('Ваш браузер: ' + name +
    '<br />Версия браузера: ' + vers +
    '<br />Кодовое название браузера: ' + code +
    '<br />Платформа: ' + platform +
    '<br />Поддержка cookie: ' + cook +
    '<br />Поддержка JavaScript: ' + je +
    '<br />userAgent: ' + ua);
</script>
```

---

<sup>1</sup> Небольшой фрагмент данных о предыстории обращений пользователя к Web-сайту, автоматически создаваемый на компьютере пользователя.

При загрузке Web-страницы браузер покажет всю информацию о себе (рис. 2.1 и 2.2).

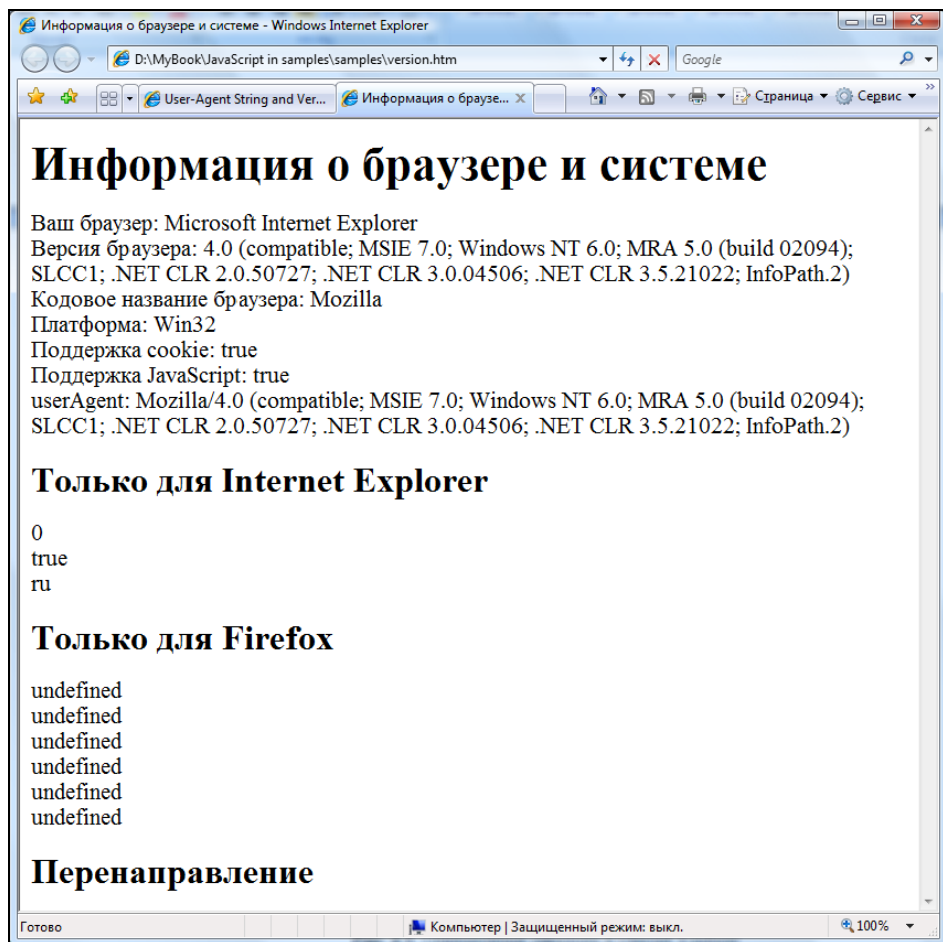


Рис. 2.1. Определение настроек в Internet Explorer

Для любознательных читателей сообщу, что Internet Explorer также поддерживает несколько своих свойств и методов:

- ☐ appMinorVersion;
- ☐ online;
- ☐ systemLanguage.

Я не стану приводить расшифровку данных свойств, поищите эту информацию в документации.

А мы напишем сценарий, предназначенный только для Internet Explorer (листинг 2.2).

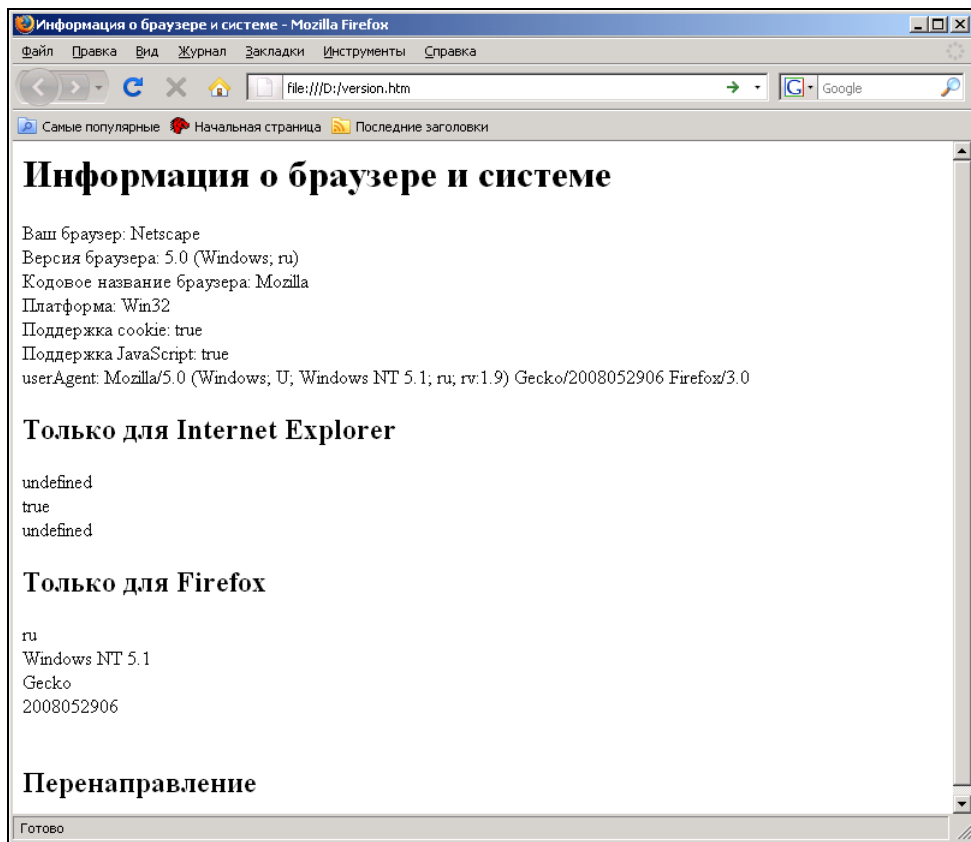


Рис. 2.2. Определение настроек в Mozilla Firefox

#### Листинг 2.2. Свойства, доступные только в Internet Explorer

```
<h2>Только для Internet Explorer</h2>
<script type = "text/javascript">
    document.write(navigator.appMinorVersion +
```

```
'<br />' + navigator.onLine +  
'<br />' + navigator.systemLanguage);  
</script>
```

А теперь приведу список свойств, присущих только Mozilla Firefox:

- ☐ language;
- ☐ oscpu;
- ☐ product;
- ☐ productSub;
- ☐ vendor;
- ☐ vendorSub.

А теперь сценарий, предназначенный для Mozilla Firefox (листинг 2.3).

### Листинг 2.3. Свойства, работающие только в Mozilla Firefox

```
<h2>Только для Firefox</h2>  
<script type = "text/javascript">  
    document.write(navigator.language +  
        '<br />' + navigator.oscpu +  
        '<br />' + navigator.product +  
        '<br />' + navigator.productSub +  
        '<br />' + navigator.vendor +  
        '<br />' + navigator.vendorSub);  
</script>
```

Как видите, различия у браузеров довольно значительны. А ведь мы рассмотрели только один объект `navigator`. С другими объектами такая же история.

## 2.3. Определение браузера

Итак, мы увидели, что каждый браузер имеет свои особенности. А так ли это важно? Очень важно. Загрузив Web-страницу со сценарием, в котором со-

держатся не поддерживаемые браузером свойства, пользователь попросту не увидит ваш замысел. Существуют два подхода к решению проблемы совместимости браузеров. Первый подход немного утомительный. Разработчик пишет два варианта для разных браузеров и затем, на основе полученной информации, направляет пользователя на нужную Web-страницу. Вот небольшой пример такого сценария (листинг 2.4).

#### Листинг 2.4. Перенаправление пользователя на нужную Web-страницу

```
<script type = "text/javascript">
  var ver = navigator.appVersion;
  if (ver.indexOf("MSIE") != -1)
  {
    alert ("У вас запущен Internet Explorer");
    // window.location.href="ie.htm";
  }
  else
    alert ("У вас запущен Firefox");
    // window.location.href="firefox.htm";
</script>
```

#### **ПРИМЕЧАНИЕ**

Готовые примеры находятся в файле Chap02\version.htm.

Я привел упрощенный сценарий, сделав допущение, что в мире существуют только два браузера. В реальной жизни, конечно, потребуется доработка; кроме того, у рассмотренного сценария есть один недостаток — требуется создание дополнительных Web-страниц под разные браузеры. Второй подход — использование оператора условия `if`. Рассмотрим на примере, для чего воспользуемся свойством `appName`. У Internet Explorer при вызове данного свойства возвращается строка `Microsoft Internet Explorer`, у браузера Mozilla Firefox — `Netscape` (листинг 2.5).

#### Листинг 2.5. Другой способ определения браузера

```
<script type = "text/javascript">
  var errorText = 'Не могу определить глубину цвета';
```

```
if (navigator.appName.indexOf('Netscape') != -1)
  if (navigator.appVersion.substr(0, 1) > 3)
    document.write('Глубина цвета: ' + window.screen.pixelDepth)
  else
    document.write(errorText);

if (navigator.appName.indexOf('Microsoft') != -1)
  if (navigator.appVersion.substr(0, 1) > 3)
    document.write('Глубина цвета: ' + window.screen.colorDepth)
  else
    document.write(errorText);
</script>

<NOSCRIPT>
  Увы, ваш браузер не поддерживает выполнение сценариев,
  поэтому я не могу определить настройки вашего монитора.
</NOSCRIPT>
```

В этом примере сначала определяется название браузера, затем его версия, после чего, с помощью поддерживаемых свойств, выводится информация. Как видно из этого сценария, у Netscape Navigator 4.0 для определения глубины цвета монитора используется свойство `pixelDepth`, а у Internet Explorer — свойство `colorDepth`.

### **ПРИМЕЧАНИЕ**

Хорошая новость для разработчиков. Браузер Mozilla Firefox поддерживает оба эти свойства.

Обратите внимание, что нам нет острой необходимости помнить всю возвращаемую строку. С помощью функции `indexOf` мы определяем позицию в строке одного слова. Этого вполне достаточно, чтобы сценарий был работоспособным. Только не забывайте, что не только Mozilla Firefox имеет имя приложения Netscape. Существует, к примеру, еще и браузер Netscape Navigator.

### **ПРИМЕЧАНИЕ**

Пример сценария находится в файле Chap02\version2.htm.

Хотя мы с вами договорились, что в этой книге все примеры тестируются только на Internet Explorer 7.0 и Mozilla Firefox 3.0.x, я все-таки приведу небольшой пример определения версии используемого браузера. Немного теории. Написать полностью универсальную функцию, определяющую браузер, установленный у пользователя, невозможно. Слишком много их развелось, и у каждого разработчика браузера свое видение объектной модели. Можно остановиться только на самых популярных и пренебречь остальными.

Итак, Internet Explorer используют в своей модели конструкции типа `document.all`, а Netscape Navigator — `document.layers`. Обе эти конструкции не являются стандартом, поэтому избегайте их использования на своих Web-страницах. Старые версии браузера Opera тоже грешили нестандартной конструкцией `document.all.item`. Но последние версии всех этих браузеров сделали шаг вперед и поддерживают правильную конструкцию `document.getElementById` (Mozilla Firefox изначально пытался следовать стандартам). Поэтому во многих сценариях используется приблизительно такой код (листинг 2.6).

#### Листинг 2.6. Еще один пример определения браузеров

```
var bNN4 = document.layers;  
var bIE4 = document.all && document.all.item;  
// проверка поддержки document.all.item необходима для отсеечения Opera  
var bW3CDOM = document.getElementById;  
var bDOMBrowser = bNN4 || bIE4 || bW3CDOM;
```

Я привел только маленький кусочек кода сценария. Применение этого кода в реальной программе останется вам в качестве домашнего задания.

## 2.4. Свойства экрана

В этом разделе мы рассмотрим параметры, позволяющие определить некоторые свойства экрана.

- `availHeight`, `availWidth` — позволяют узнать рабочую высоту и ширину экрана в пикселах без учета панели задач и других панелей, постоянно занимающих место на экране, например, панели MS Office или ICQ.
- `height`, `width` — позволяют вычислить высоту и ширину экрана.



К счастью, эти параметры поддерживаются обоими нашими браузерами. Вот пример, позволяющий вычислить доступные размеры для окна браузера и раскрывающий его в соответствии с полученными результатами (листинг 2.7).

#### Листинг 2.7. Развернуть окно на максимально возможные размеры

```
<script type = "text/javascript">
    function fullWindow()
    {
        window.moveTo(0, 0);
        window.resizeTo(screen.availWidth, screen.availHeight);
    }
</script>

</head>
<body onload="fullWindow()">
```

#### ПРИМЕЧАНИЕ

Пример сценария находится в файле Chap02\fullscreen.htm.

Раз уж мы можем узнать размеры монитора пользователя, то можем создавать при необходимости разные Web-страницы под различные разрешения монитора. Вот самый простой сценарий, который приходит в голову (листинг 2.8).

#### Листинг 2.8. Определение разрешения

```
<script type = "text/javascript">
    var msg="У вашего монитора очень высокое разрешение";

    if (screen.width == 640 || screen.height == 480){
        location=("640x480.htm");}
    else if (screen.width == 800 || screen.height == 600){
        location=("800x600.htm");}
    else if (screen.width == 1024 || screen.height == 768){
        location=("1024x768.htm");}
```

```
else if(screen.width == 1152 || screen.height == 864){
    location=("1152x864.htm");}
else if (screen.width == 1280 || screen.height == 720){
    location=("1280x720.htm");}
else if (screen.width > 1280 || screen.height > 720){
    alert(msg);
}
</script>
```

Вам придется заранее подготовить пять разных Web-страниц под каждое из популярных разрешений монитора: 640x480.htm, 800x600.htm, 1024x768.htm, 1152x864.htm и 1280x720.htm.

### **ПРИМЕЧАНИЕ**

Пример сценария, определяющего разрешение экрана, находится в файле Chap02\screen.htm.

## **2.5. Размеры документа**

Кроме размеров экрана, иногда требуется знать размеры самого документа в браузере. Здесь тоже приходится писать разный код для браузеров, т. к. используются различные свойства (листинг 2.9).

### **Листинг 2.9. Получение размеров документа в браузере**

```
<script type = "text/javascript">
var W; // ширина документа в браузере
var H; // высота документа в браузере

function checkSize()
{
    if(document.all)
    {
        W = document.body.clientWidth;
        H = document.body.clientHeight;
    }
}
```

```
else
{
    W = innerWidth;
    H = innerHeight;
}
}
</script>
// Вызываем функцию в нужном месте для получения размеров документа
<script type = "text/javascript">
    checkSize();
    alert("Высота: " + H);
    alert("Ширина: " + W);
</script>
```

### **ПРИМЕЧАНИЕ**

Пример сценария, определяющего размер документа в браузере, находится в файле Chap02\browsersize.htm.

## **2.6. Информация об операционной системе**

Чтобы получить информацию об операционной системе посетителя вашей Web-страницы, можно воспользоваться свойством `appVersion` объекта `navigator` (листинг 2.10).

### **Листинг 2.10. Получение информации об операционной системе**

```
function getOS()
{
    if (navigator.appVersion.indexOf('Windows')>=0) return 'Windows';
    if (navigator.appVersion.indexOf('Linux')>=0)   return 'Linux';
    if (navigator.appVersion.indexOf('Sun')==0)      return 'SunOS';
}
```

Вы можете самостоятельно расширить список известных операционных систем. В приведенном примере я ограничился тремя наиболее известными операционными системами.

## 2.7. Свойство *userAgent*

Давайте подробнее рассмотрим свойство `userAgent` объекта `navigator`, с помощью которого можно извлечь много полезной информации. Когда вы посещаете Web-страницу, то ваш браузер посылает строку `userAgent` серверу, по которому сервер определяет ваш браузер, номер версии и другие детали о вашей системе. Данная информация помогает серверу подстроиться под работу вашего браузера. Строка, возвращаемая свойством `userAgent`, у всех браузеров уникальная. Например, у браузера Internet Explorer 7.0, запущенного в системе Windows Vista, свойство возвращает очень длинную строку

```
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; MRA 5.0 (build 02094);  
SLCC1; .NET CLR 2.0.50727; .NET CLR 3.0.04506; .NET CLR 3.5.21022;  
InfoPath.2)
```

### ПРИМЕЧАНИЕ

Чтобы быстро получить строку `userAgent` в своем браузере, наберите прямо в адресной строке следующее (с соблюдением регистра символов): `javascript:alert(navigator.userAgent)`.

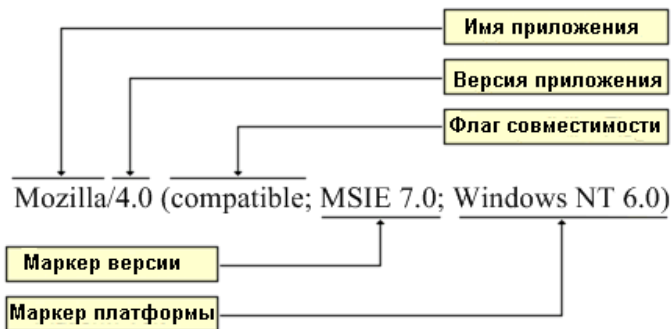


Рис. 2.3. Строка `userAgent`