

Майкл Моррисон

Изучаем

JavaScript



Улучшай
качество
взаимодействия
пользователя
с веб-страницей



Научись
оптимизировать
JavaScript-код



Избавься от страха
перед обработчиком
событий



Освой концепцию
и синтаксис JavaScript
максимально эффективно



Управляй
HTML-кодом
с помощью
DOM



Проверяй свои знания
с помощью сотен
упражнений и примеров

O'REILLY®

ПИТЕР®

Head First JavaScript

Wouldn't it be dreamy if there was a way to learn JavaScript from a book without wanting to set fire to it halfway through and swearing off the Web forever? I know, it's probably just a fantasy...



Michael Morrison

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

W.870 &
08M

Изучаем JavaScript

Как было бы здорово изучить JavaScript, не испытывая желания бросить все на половине пути и никогда больше не заходить в Интернет! Наверное, об этом можно только мечтать...

Майкл Моррисон



Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Ростов-на-Дону · Екатеринбург · Самара · Новосибирск
Киев · Харьков · Минск
2012

ПЕЧАТНИЦА
Терского

Краткое содержание

	Введение	23
1	Интерактивная сеть: <i>Реакции виртуального мира</i>	35
2	Хранение данных: <i>Все на своем месте</i>	65
3	Исследование клиента: <i>Знакомство с браузером</i>	115
4	Принятие решений: <i>Если на дороге развилка...</i>	163
5	Циклы: <i>Рискуя повториться</i>	215
6	Функции: <i>Множественное использование</i>	267
7	Формы и проверка данных: <i>Пусть он все расскажет</i>	311
8	Управление страницами: <i>Управление HTML с DOM</i>	363
9	Оживляем данные: <i>Объекты как франкенданные</i>	411
10	Специальные объекты: <i>Работа со специальными объектами</i>	465
11	Охота на ошибки: <i>Когда сценарий не работает</i>	499
12	Динамические данные: <i>Удобные веб-приложения</i>	549

Содержание

Введение

Ваш мозг думает о JavaScript. Вы сидите за книгой и пытаетесь что-нибудь *выучить*, но ваш мозг продолжает считать, что вся эта писанина не важна. Ваш мозг говорит: «Выгляни в окно! На свете есть более важные вещи. Например, серфинг или голодный тигр, когда ты попался на его пути». Как *заставить* ваш мозг думать, что ваша жизнь действительно зависит от знания JavaScript?

Для кого написана эта книга?	24
Мы знаем, о чем вы думаете	25
Метапознание: наука о мышлении	27
Как заставить мозг повиноваться?	29
Примите к сведению	30
Технические редакторы	32
Благодарности	33

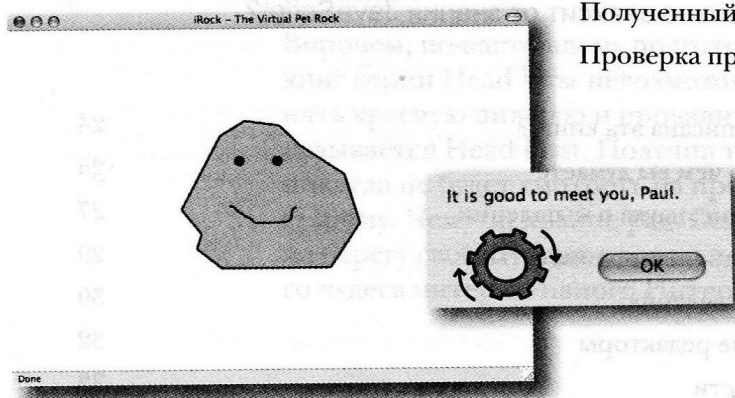
интерактивная сеть

1

Реакции виртуального мир

Устали представлять Интернет набором пассивных страниц? Кто из нас не держал в руках книгу. Их читаешь, в них находишь информацию. Но они не **интерактивны**. Как и интернет-страницы без JavaScript. Без сомнения, отправить данные формы и проделать другие трюки можно и при помощи кода HTML и CSS, но реальная **интерактивность** требует **более умного подхода** и большей работы... зато и результат впечатляет **намного больше**.

То, что нужно людям	36
И ничего... как будто говоришь со стенкой	37
А JavaScript отвечает	38
Свет, камера, взаимодействие!	40
Тег <script>	45
Ваш браузер понимает HTML, CSS И JavaScript	46
Помоги виртуальному другу человека	49
Сделайте iRock интерактивным	50
Веб-страница iRock	51
Тестирование	51
События	52
Оповещение пользователей	53
iRock приветствует вас	54
Сделайте объект iRock действительно интерактивным	56
Взаимодействие должно быть ДВУСТОРОННИМ	57
Как узнать имя пользователя	58
Полученный результат	61
Проверка приложения iRock 1.0	62

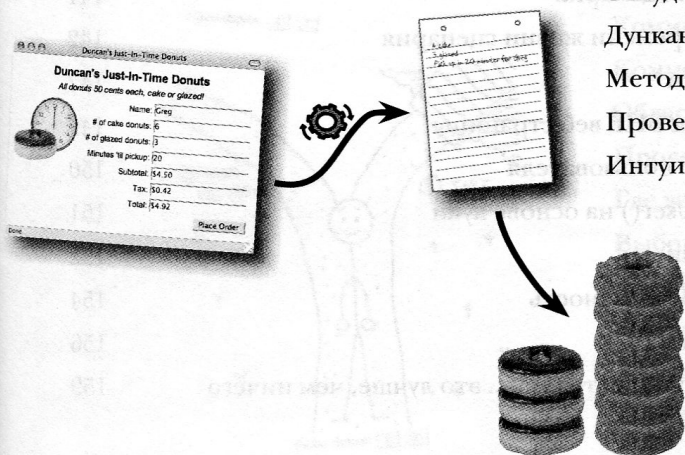


2 **Хранение данных**

Все на своем месте

В реальном мире люди часто не придают значения местам для хранения своего имущества. В JavaScript такое поведение невозможно. Ведь там не существует роскоши в виде огромных шкафов и гаражей на три машины. В JavaScript **все имеет свое место**, и ваша задача в этом убедиться. Мы поговорим о **данных** — как их **представить**, как **хранить их** и как их **найти** после сохранения. Вы научитесь превращать захламленные комнаты с данными в аккуратные помещения с ящиками, каждый из которых имеет пометку.

	Сохранение данных	66
	Типы данных	67
	Константы и переменные: постоянное и изменяемое	72
	Исходное состояние переменных	76
	Присвоение значений	77
	Упрямые константы	78
	Что в имени тебе моем?	82
	Корректные и некорректные имена	83
	СтильВерблюда	84
	Следующий этап	87
	Планируем веб-страницу	88
	Инициализируйте данные... или...	93
	NaN — это НЕ число	94
	Складывать можно не только числа	96
	Методы parseInt() и parseFloat()	97
	Откуда берутся лишние пончики?	98
	Дункан обнаруживает шпиона	102
	Метод getElementById()	103
	Проверка данных формы	104
	Интуитивный ввод данных	109



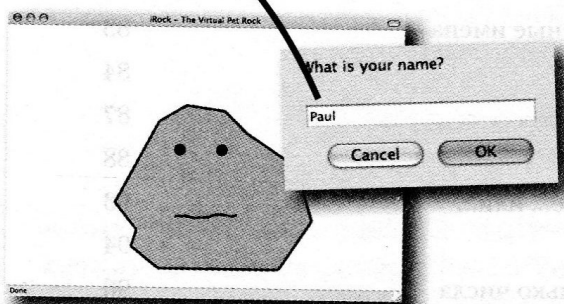
Исследование клиента

Знакомство с браузером

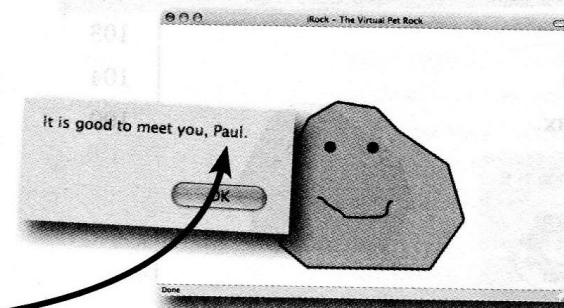
Иногда JavaScript хочет знать, что происходит в окружающем мире. Ваши сценарии могут существовать в качестве кода на веб-страницах, но по большей части они живут в мире, создаваемом браузером или клиентом. **Умным сценариям** часто необходимо знать больше о мире, в котором они живут, в этом случае они могут **общаться с браузером**, чтобы узнать про него как можно больше. Независимо от того, что требуется узнать: размер экрана или нажата ли кнопка в браузере, они постоянно поддерживают отношения с браузером.

Клиент, сервер и JavaScript	116
Что браузер может сделать для вас?	118
Объект iRock слишком счастлив	119
Таймеры	122
Как работает таймер	123
Метод setTimeout()	124
Анализ метода setTimeout()	125
Зависимость от размера экрана	129
Ширина окна браузера	130
Задание ширины окна	131
Высота и ширина объекта iRock	132
iRock должен соответствовать странице	133
Событие onresize	137
Событие onresize для камешка	138
Мы уже встречались?	140
Время жизни сценария	141
Продление времени жизни сценария	142
Свойства куки	147
Код JavaScript ВНЕ веб-страницы	149
Приветствие пользователя	150
Метод greetUser() на основе куки	151
Создание куки	152
Влияние на безопасность	154
Мир без куки	156
Разговор с пользователями... это лучше, чем ничего	159

Start here!



Finish!



Принятие решений

4

Если на дороге развилка...

Жизнь неотделима от принятия решений. Стоять или идти, пойти на сделку с негодяем или пойти в суд... Результата невозможно добиться без выбора. То же самое происходит в JavaScript — **вам приходится выбирать между различными вариантами сценария. Приходится то и дело принимать решения.** Стоит ли поверить данным, введенным пользователем, и отправить его охотиться на львов? Или же проверить еще раз, может быть, он всего лишь пытался заказать билет до Львова? Выбор за вами!

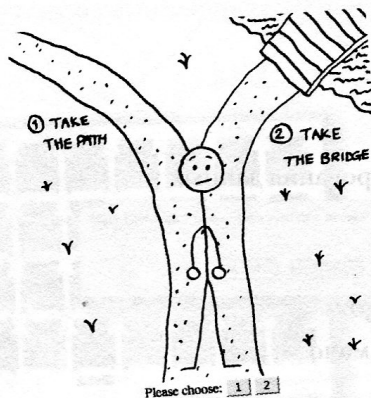
Счастливчик, спускайся ко мне!	164
«Уесли» так, то сделай что-нибудь	166
Оператор if	167
Когда вариантов два	169
Вы можете сделать множественный выбор	170
Ключевое слово else	171
Переменные как двигатель истории	174
Недостающие части истории	175
Совмещение усилий	176
Запись при помощи if/else	182
Вложенный оператор if	183
Управление при помощи методов	185
Псевдокод	186
Проблемы нарисованного человечка	190
!= т-с-с-с, мне нечего тебе сказать...	191
Операторы сравнения	192
Комментарии	194
Комментарии начинаются с //	195
Область видимости	197
Проверим область видимости	198
Где живут мои данные?	199
Выбор из пяти	202
Переусложнение конструкции	203
Оператор switch/case	205
Анализ оператора switch	206
Тест-драйв нового варианта «Приключений»	211

Welcome to STICK FIGURE ADVENTURE



Click either
button to
start...

Please choose: 1 2



Please choose: 1 2

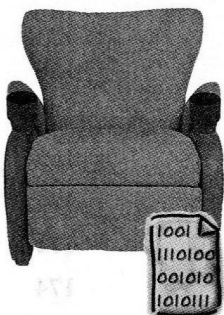
Циклы

5

Рискуя повториться

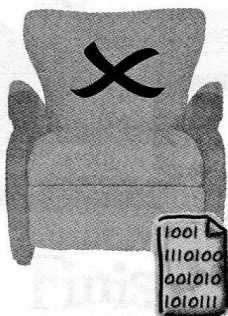
Говорят, что повторение — мать учения. Заниматься новыми и интересными делами здорово, но наши дни, как правило, состоят из рутины. Доведенное до автоматизма мытье рук, нервный тик, щелчок на кнопке Reply To All при получении любого дурацкого сообщения! Кажется, повторение не самая лучшая вещь в этом мире. А вот в мире JavaScript без него никак. Вы удивитесь, как часто бывают востребованы одни и те же фрагменты кода. Здесь вам на помощь приходят циклы. Без них пришлось бы снова и снова набирать один и тот же код.

Available



seat_avail.png

Unavailable



seat_unavail.png

Select



seat_select.png

Место помечено крестом	216
И снова дежавю... цикл for	217
Охота за сокровищами с циклом for	218
Составные части цикла for	219
Специальные места для мачо	220
Проверка доступности мест	221
Циклы, HTML и свободные кресла	222
Места, как переменные	223
Массивы	224
Значения сохраняются с ключами	225
От JavaScript к HTML	229
Визуализация кресел	230
Проверка	235
Бесконечные циклы	236
Условие выхода из цикла	237
Прерывание действия	238
Логические операторы	244
Цикл while	248
Анализ цикла while	249
Выбор подходящего цикла	251
Кинотеатр — место моделирования данных	257
Двумерные массивы	258
Два ключа доступа	259
Двумерная версия Mandango	261
Целый кинотеатр мест для мачо	264

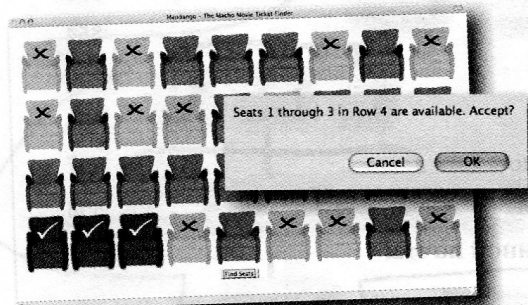
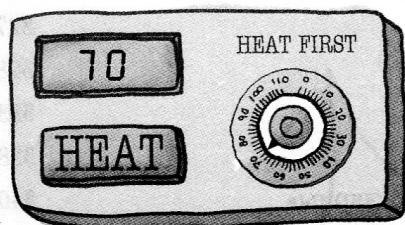
Функции

6

Множественное использование

Начни JavaScript выступать за экологию, это выступление возглавили бы функции. Ведь именно они увеличивают эффективность кода и позволяют использовать его многократно. Они ориентированы на решение задач и позволяют все систематизировать. Функции дают возможность упростить любой сценарий, ну кроме разве что и так простых. Их значение невозможно оценить, поэтому просто скажем, что именно функции делают сценарии такими экологичными.

Источник всех проблем	268
Функции как способ решения	270
Из чего состоит функция	271
Уже знакомые вам функции	272
Улучшаем наш термостат	275
Передача информации функциям	276
Аргументы как данные	277
Избавляемся от дублирующегося кода	278
Функция, задающая места	281
Функция setSeat()	283
Обратная связь	285
Возврат данных	286
Возвращаемые значения	287
Информация о статусе места	291
Отображение статуса	292
Связь функции с изображением	293
Дублирующий код	294
Отделите функциональность от содержимого	295
Функции — это тоже данные	296
Вызов функции и ссылка на нее	297
События, обратный вызов и атрибуты HTML	301
Ссылки на функции	302
Литерал функции	303
А где же связывание?	304
Оболочка HTML-страницы	307



Формы и проверка данных

7 Пусть он все расскажет

Для получения информации от пользователей при помощи JavaScript вам не потребуется быть джентльменом. Но вы должны быть аккуратны. Люди часто делают ошибки, а это означает, что данные, полученные при помощи веб-форм, далеко не всегда **корректны**. Проверая вводимые данные при помощи JavaScript, вы увеличиваете **надежность веб-приложений** и снимаете дополнительную нагрузку с серверов. Полоса пропускания нам пригодится для восхитительных видеороликов и чудесных фотографий.

HTML-форма фирмы Bannerocity	313
Когда языка HTML недостаточно	314
Доступ к данным формы	315
Цепочка событий	317
Событие onBlur	318
Сообщение проверки	319
Проверка полей на наличие данных	323
Проверка без предупреждающих всплывающих окон	324
Усложняем наш валидатор	325
Размер имеет значение...	327
Проверка длины	328
Проверка индексов	333
Проверка даты	338
Регулярные выражения не «регулярны»	340
Задание шаблона	341
Метасимволы	343
Количество повторений	344
Проверка данных при помощи регулярных выражений	348
Диапазон вхождений	351
Выбери это... или то	353
Никаких случайностей	354
Вы меня слышите?	355
Вам письмо	356
Исключение — это правило	357
Дополнительные символы	358
Проверка адреса электронной почты	359

Bannerocity...banner ads in the sky!

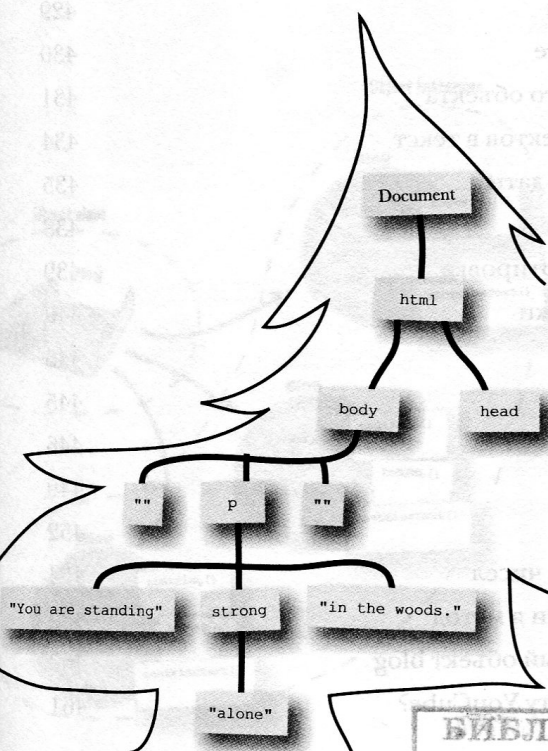
Управление страницами

8

Управление HTML с DOM

Управление содержимым веб-страницы при помощи JavaScript напоминает приготовление еды. Конечно, это не настолько грязное занятие... И, увы, вы не сможете съесть результат своих трудов. Тем не менее вы получаете **полный доступ к HTML-ингредиентам**, из которых состоит веб-страница, и, что еще важнее, вы можете **менять** исходный рецепт. Ведь **JavaScript** дает возможность **управлять HTML-кодом веб-страницы**, что открывает для вас целый ряд интереснейших перспектив, которые реализуются посредством **набора стандартных объектов DOM**.

Функциональный, но неудобный	364
Без всплывающих окон	365
Доступ к HTML-элементам	367
Внутренний код элемента	368
Объектная модель документа (DOM)	373
Страница как набор узлов	374
Свойства узлов	377
Редактирование текста	380
«Приключение», совместимое со стандартами	385
Проектирование лучше, варианты чище	387
И снова замена текста в узлах	388
Функция, заменяющая текст узла	389
Динамические параметры	390
Интерактивность	391
Значение стиля: CSS и DOM	392
Замена классов стилей	393
Стильные варианты	394
Проверка работы приложения	395
Пустая кнопка	396
Настройка «а ля style»	397
Без фиктивных кнопок	399
Усложняем «Приключения»	400
Поход по дереву решений	402
Превратим историю в HTML	403
Обработка HTML-кода	404
Отслеживание «Приключений»	407



Оживляем данные

9

Объекты как франкенданные

Объекты JavaScript вовсе не так ужасны, как заставил вас думать доктор. Зато они интересны тем, что соединяют друг с другом отдельные части языка JavaScript, делая его более мощным. **Объекты объединяют данные с действиями** в новый тип, намного более «живой», чем все, что вы использовали раньше. Вы познакомитесь с **массивами, которые сортируют себя сами**, со строками, которые умеют искать в своем составе указанные последовательности символов, и многими другими замечательными особенностями.

Вечеринка в стиле JavaScript	412
Данные + действия = объект	413
Данные – это свойство объекта	414
Ссылка на члены объекта	415
Специальные объекты	419
Конструктор	420
Структура конструктора	421
Создание объектов blog	422
Необходимость сортировки	427
Объект для дат	428
Вычисление времени	429
Пересмотр дат в блоге	430
Объект внутри другого объекта	431
Преобразование объектов в текст	434
Доступ к фрагментам даты	435
Массивы как объекты	438
Пользовательская сортировка	439
Упрощение сортировки	440
Поиск по массиву	443
Метод indexOf()	445
Поиск по блогу	446
Поиск заработал!	449
Объект Math	452
Генерация случайных чисел	454
Превращение функции в метод	459
Восхитительный новый объект blog	460
Что дают объекты блогу YouTube?	461

Data

```
var who;
var what;
var when;
var where;
```

+

Actions

```
function display(what, when, where) {
    ...
}
function deliver(who) {
    ...
}
```

Object

=

```
var who;
var what;
var when;
var where;
```

```
function display() {
    ...
}
function deliver() {
    ...
}
```

10

Специальные объекты

Работа со специальными объектами

Если бы все было так легко, мы бы, конечно, так и сделали. JavaScript не гарантирует возврат денег, но вы действительно можете делать с ним все, что захотите. Специальные объекты — это эквивалент тройного эспрессо с сахаром и корицей. Вот такая специальная чашка кофе! Точно так же в специальных объектах вы можете смешивать код, добываясь именно того результата, который вам нужен, и пользуясь преимуществами свойств и методов. И в конце получается объектно-ориентированный код, расширяющий язык JavaScript... только для вас!

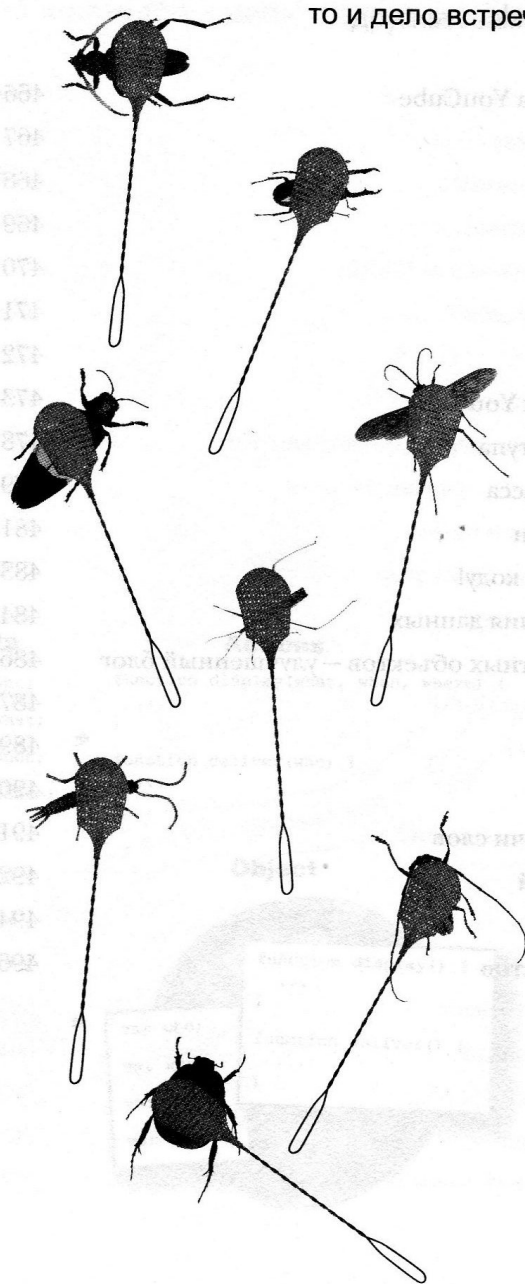
Object class	Снова о методах блога YouTube	466
Object instances	Перегрузка методов	467
Blog	Классы и реализации	468
Blog	Реализации	469
Blog	Ключевое слово this	470
Blog	Методы классов	471
Blog	Прототипы	472
Blog	Классы, прототипы и YouTube	473
Blog	Свойства общего доступа	478
Blog	Создание свойств класса	479
Blog	Подписан и доставлен	481
Blog	Нет дублирующемуся коду!	483
Blog	Метод форматирования данных	484
Blog	Расширение стандартных объектов — улучшенный блог	486
Blog	Методы классов	487
Blog	Функция сравнения	489
Blog	Вызов метода класса	490
Blog	Картинка стоит тысячи слов	491
Blog	Вставка изображений	492
Blog	Добавление галереи	494
Blog	Блог на основе объектов	496

Охота на ошибки

11

Когда сценарий не работает

Даже самые лучшие планы в JavaScript иногда не реализуются. И когда это происходит, главное — не паниковать. Лучшие программисты не те, которые никогда не делали ошибок, — на самом деле это просто лгуны. Лучшие — это те, кто может успешно обнаружить и устранить ошибку. Отладчики высокой квалификации **нарабатывают хорошую манеру написания кода**, минимизирующую вероятность появления неприятных ошибок. **Лучше предотвратить, чем потом бороться.** Тем не менее ошибки то и дело встречаются, и вам нужен арсенал средств борьбы с ними...



Устранение дефектов	500
Проблемы с калькулятором для IQ	501
Различные баузеры	502
Несложная отладка	505
Неопределенные переменные	509
Работа с цифрами	511
Звонки на радио	512
Начинаем расследование	513
Проверка синтаксиса (ошибка #1)	514
Аккуратнее со строками	515
Кавычки и апострофы	516
Esc-символы	517
Неопределенность функции (Ошибка #2)	518
Побеждают все (Ошибка #3)	520
Отладка с помощью всплывающих окон	521
Следим за значением переменной	522
Некорректная логика	524
Проигрывают все! (Ошибка #4)	528
Атака всплывающих окон	529
Пользовательская консоль	531
Самая противная ошибка	538
Три самых популярных типа ошибок	539
Комментарии	542
Дважды объявленные переменные	544

12

Динамические данные

Удобные веб-приложения

Современный Интернет очень отзывчив, страницы умеют реагировать на каждый каприз пользователя. Именно об этом мечтают многие разработчики. JavaScript играет важную роль в осуществлении этой мечты при помощи технологии **Аjax**, позволяющей эффективно менять «чувствительность» страниц. Благодаря Ajax страницы научились быстро загружаться и динамически сохранять данные, отвечая на действия пользователя в реальном времени без необходимости перезагрузки браузера.

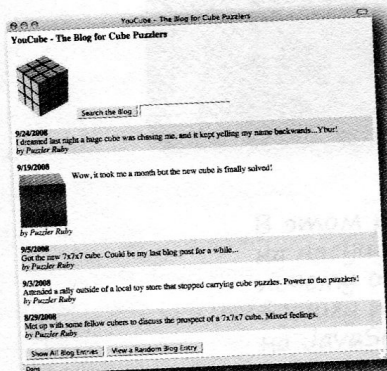
```
<html>
<head>
...
</head>
<body>
...
</body>
</html>
```

youcube.html



```
<blog>
<title>...
<author>...
<content>
<entry>
...
</entry>
...
</entries>
</blog>
```

blog.xml



Жажда динамических данных	550
Блог, управляемый данными	551
Аjax как возможность для общения	553
Форматирование с помощью XML	555
XML + HTML = XHTML	557
XML и данные блога YouCube	559
Добавим к блогу Ajax	562
Интерфейс XMLHttpRequest	564
Запрос с объектом XMLHttpRequest	567
Анализ запросов Ajax	571
Создание запросов	575
Закончишь — вызови меня	576
Обработка ответа	577
DOM как выход из положения	578
YouCube, управляемый данными	583
Неработающие кнопки	585
Кнопкам нужны данные	586
Функция, экономящая время	589
Запись данных в блог	590
Требования PHP	593
Данные для PHP-сценария	594
Отправка данных на сервер	597
Делаем работу с блогом еще удобнее	602
Автозаполнение полей	603
Повторяющаяся задача?	604