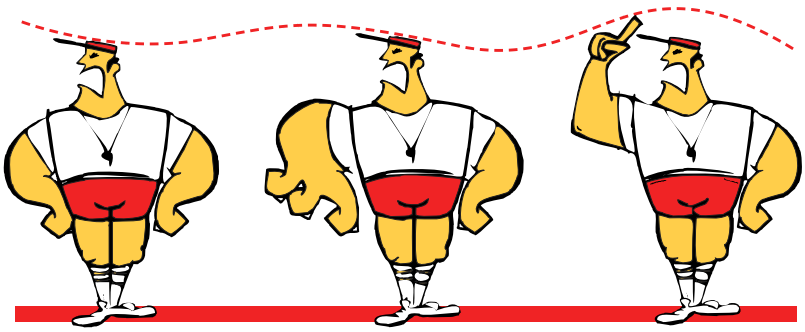
Мастера FLASH

Роберт Пеннер

ПРОГРАММИРОВАНИЕ ВО FLASH MX



СИМВОЛ®

Robert Penner's Programming Macromedia Flash™ MX

Robert Penner

McGraw-Hill/Osborne

Мастера FLASH

Программирование во Flash MX

Роберт Пеннер

Санкт-Петербург — Москва
2005

Серия «Мастера FLASH»

Роберт Пеннер

Программирование во Flash MX

Перевод С. Иноземцева

Главный редактор	*А. Галунов*
Зав. редакцией	*Н. Макарова*
Научный редактор	*М. Антипин*
Редактор	*А. Лосев*
Художник	*В. Гренда*
Корректор	*С. Доничкина*
Верстка	*Н. Гриценко*

Пеннер Р.
Программирование во Flash MX. – Пер. с англ. – СПб: Символ-Плюс, 2005. – 432 с., ил.
ISBN 5-93286-072-3

Роберт Пеннер – известный новатор, пишущий на языке ActionScript, – делится своим глубоким пониманием сути программирования в Macromedia Flash. Он обладает талантом объединять сложные понятия математики и физики с чистой эстетикой графики Macromedia Flash. Автор проведет вас от простейших приемов проектирования и кодирования движения до способов разработки профессионального объектно-ориентированного кода, позволяющего создавать интерактивность, цвет, звук и движение. Эта книга раздвинет границы вашего воображения. Оригинальные идеи автора, создавшего танцующий фрактал, имитацию снежной бури и северного сияния, подвигнут вас к реализации интересных динамических проектов и анимации в Macromedia Flash на мощном и гибком языке ActionScript.

Вот мнение Колина Мука, автора бестселлеров по ActionScript: «Этой книге суждено стать классической работой по программированию графики в Macromedia Flash. Flash-разработчики получили фундаментальные и практичные образцы объектно-ориентированного программирования, о которых они мечтали много лет. Как новички, так и эксперты в области ООП высоко оценят стиль Пеннера.»

ISBN 5-93286-072-3
ISBN 0-07-222356-1 (англ)

© Издательство Символ-Плюс, 2005
Authorized translation of the English edition © 2002 McGraw-Hill/Osborne. This translation is published and sold by permission of McGraw-Hill Companies, the owner of all rights to publish and sell the same.

Все права на данное издание защищены Законодательством РФ, включая право на полное или частичное воспроизведение в любой форме. Все товарные знаки или зарегистрированные товарные знаки, упоминаемые в настоящем издании, являются собственностью соответствующих фирм.

Издательство «Символ-Плюс». 199034, Санкт-Петербург, 16 линия, 7,
тел. (812) 324-5353, edit@symbol.ru. Лицензия ЛП N 000054 от 25.12.98.
Налоговая льгота – общероссийский классификатор продукции
ОК 005-93, том 2; 953000 – книги и брошюры.
Подписано в печать 06.06.2005. Формат 70x100 1/16 . Печать офсетная.
Объем 27 печ. л. Тираж 2000 экз. Заказ N
Отпечатано с готовых диапозитивов в ГУП «Типография «Наука»
199034, Санкт-Петербург, 9 линия, 12.

Вильяму Рональду Пеннеру и Вильме Джун Пеннер, которые заставляли меня «погулять и подышать свежим воздухом».

Об авторе

Роберт живет в Канаде в г. Ванкувер. Он свободный Flash-разработчик, консультант и лектор, чьи интересы сосредоточены в области математики и объектно-ориентированного проектирования. Модератор и один из основных авторов сайта Ultrashock.com, Роберт известен во всем мире своей технической изобретательностью и органичным стилем. Он предложил такие новшества, как интеграция кнопки возврата во Flash и математические уравнения для реализации ускорения. Роберт не ограничивается модерированием нескольких форумов на Ultrashock и горячими спорами об ООП с членами сообщества Flashcoders на сайте Chattyfig. Сотрудничество с Axis-media.com принесло ему две номинации на фестивалях Flash-фильмов. Законченные проекты и эксперименты Роберта можно увидеть на сайте *http://www.robertpenner.com*.

Оглавление

Введение

Больше года тому назад Джим Шахтерль (Jim Schachterle) предложил мне написать книгу о Flash для издательства Osborne/McGraw-Hill. У него был весьма необычный замысел: проект должен быть «на пересечении программирования и дизайна». Иными словами, речь в книге должна идти о создании визуальных элементов с помощью кода. К этому моменту я уже некоторое время занимался чем-то подобным и держал на своем сайте (*www.robertpenner.com*) ряд экспериментальных анимаций, выполненных просто для развлечения.

Джим был твердо убежден, что книга должна давать концепции и принципы, которые можно применять самыми разными способами, а не быть просто перечнем работ. Я с радостью согласился. В моей голове вертелось множество идей, о которых хотелось написать, и подобный подход позволял провести достаточно глубокое обсуждение темы.

Итак, эта книга – концептуальное изложение моего подхода к созданию динамической компьютерной графики на языке ActionScript. Ее основными темами являются объектно-ориентированное программирование, тригонометрия, системы координат, векторы и программирование, управляемое событиями. Разобравшись в этих вопросах достаточно хорошо, можно переходить к обсуждению движения, законов физики, а также к окрашиванию и рисованию фигур. Я постараюсь разъяснить читателю основополагающие принципы этих разделов математики и физики и способы реализации их на языке ActionScript. Несомненно, Flash и ActionScript будут развиваться в ближайшие годы, и код, с помощью которого создается анимация, будет меняться вместе с ними. Однако навыки, приобретенные при изучении этой книги, и понимание вечных принципов математики и законов движения помогут читателю перенести код в новый контекст.

О чем эта книга

В части I «Начало» я рассказываю о том, что предшествовало моему увлечению программированием и что сформировало меня как программиста.

Глава 1 «Flash глазами энтузиаста: процесс и дисциплина» автобиографична. Я рассказываю историю моего открытия Flash и делюсь жизненным опытом, повлиявшим на мою работу. Я также излагаю некоторые личные принципы (которые называю «дисциплинами»), формирующие мою повседневную профессиональную деятельность.

Глава 2 «Объектно-ориентированное программирование» является введением в объектно-ориентированный анализ, проектирование и программирование. Она демонстрирует путь, который проходит разработчик от проектных спецификаций до объектно-ориентированного кода, а также способы реализации на языке ActionScript таких концепций объектно-ориентированного программирования, как классы, методы и наследование.

Часть II «Базовые концепции» представляет фундаментальные понятия математики и теории программирования.

Глава 3 «Математика 1: тригонометрия» закладывает математические основы анимации на языке ActionScript и содержит информацию, необходимую для понимания глав, посвященных векторам. Обсуждаются такие ключевые понятия, как углы, треугольники, теорема Пифагора и полярные координаты.

Глава 4 «Математика 2: векторы на плоскости» описывает векторы и способы их реализации на языке ActionScript. Читатель познакомится со сложением векторов, их скалярным произведением и умножением вектора на скаляр, а также увидит воплощение этих и других действий в коде.

Глава 5 «Математика 3: векторы в трехмерном пространстве» переводит обсуждение в третье измерение. Разъясняются трехмерные координаты, их проекция на плоскость экрана и поворот вокруг координатных осей, и все это сопровождается кодом на языке ActionScript.

Глава 6 «Программирование, управляемое событиями» представляет новую модель событий Flash MX и ее отличия от модели, принятой во Flash 5. Объясняется, что такое обработчики событий и слушатели. Кроме того, демонстрируется, как определять пользовательские источники событий, способные рассылать сообщения.

предыдущих глав и распространяет его на такие области, как движение, законы физики, цвет и форма.

Глава 7 «Движение, твининг и ускорение» посвящена концептуальным и математическим основам движения. Исследуются различные уравнения ускорения и разрабатывается объектно-ориентированный подход к реализации твининга в коде.

Глава 8 «Физика» разъясняет ключевые понятия физики: ускорение, трение и сила, а также свойства колебательного движения.

Глава 9 «Раскрашивание объектов средствами языка ActionScript» представляет разнообразные подходы к динамическому окрашиванию графики с помощью класса `Color` среды Flash.

Глава 10 «Рисование средствами языка ActionScript» обсуждает рисование прямых и кривых линий и заливку фигур с использованием Shape Drawing API, интерфейса, принятого во Flash MX. Демонстрируется, как применять его для рисования более сложных фигур.

Часть IV «Обзор проектов» представляет собой углубленное обсуждение четырех моих экспериментальных анимаций.

Глава 11 «Проект Aurora Borealis (Северное сияние)» описывает интерактивную двухмерную систему частиц, имитирующую северное сияние.

Глава 12 «Проект Snowstorm (Снежная буря)» посвящена трехмерной системе частиц, моделирующей полет снежинок при сильном ветре.

Глава 13 «Проект Fractal Dancer (Фрактал-танцор)» представляет читателю так называемое самоподобное[1] дерево (фрактальную структуру), реагирующее на поведение мыши и способное запоминать и воспроизводить движение.

Глава 14 «Проект Cyclone (Вихрь)» описывает систему частиц, представляющую интерактивную модель вихря.

Как читать эту книгу

Вероятно, лучше всего читать эту книгу нелинейным и интерактивным образом. Конечно, ее главы расположены в определенном логическом порядке, и излагаемые в них идеи вытекают друг из друга. Но в то же время какие-то вопросы могут стать до конца понятными лишь при повторном прочтении, когда знаний станет больше. Итак, я предлагаю читателю проделать несколько проходов по тексту для лучшего усвоения информации. При этом можно перескакивать с одного раздела на другой, так сказать, ради «перекрестного опыления» идей.

Исходные файлы

Исходные файлы, упоминающиеся в книге, доступны на сайте издательства *http://www.osborne.com* и на моем личном сайте *http://www.robertpenner.com/profmx/*.

[1] Самоподобие – одно из основополагающих понятий фрактальной геометрии. – *Примеч. науч. ред.*

Благодарности

Я благодарен

издательству Osborne/McGraw-Hill за предложение написать эту книгу;

Джиму Шахтерлю (Jim Schachterle) за блестящий проект книги, за то, что вселял в меня силы, и за его профессионализм;

Биллу Спенсеру (Bill Spencer) за веру в меня, которую он передал Джиму;

Брэндону Вильямсу (Brandon Williams), известному под псевдонимом Ahab, за его одержимость математикой;

фирме Macromedia за Flash;

Flash-сообществу за то, что в нем никогда не бывает скучно, особое спасибо Flashcoders и Ultrashock;

Энди Парку (Andy Park), Карен Кук (Karen Cook), Алану и Нехаме Вайзманам (Alan, Nechama Wiseman) и Марку Андерсону (Mark Anderson) за мудрые советы и поддержку в процессе создания книги;

Дейву, Кевину, Даррену и Вилли за постмодернистский опыт и грязные санузлы;

Создателю – за фракталы.

I

Начало

1

Flash глазами энтузиаста: процесс и дисциплина

Мы – то, что мы делаем постоянно.
Следовательно, совершенство достигается
не единичным усилием, а образом жизни.

– Аристотель

Я – приверженец Flash. Теперешний этап моей жизни посвящен исследованию возможностей Flash и их творческому осмыслению. Среда Flash необычайно интересна, и она вознаграждает своих сторонников. Она захватила мой ум и сердце, – как я могу сопротивляться?

Меня также всегда интересовало, как люди приобретают знания и умения. Поразительно наблюдать, как несколько базовых понятий и навыков открывают перед человеком целый мир. Я не могу назвать себя выдающимся преподавателем, но с увлечением делюсь своими знаниями. Поскольку я хочу, чтобы мой труд приносил плоды, я стараюсь подавать материал просто и понятно. Возможно, читателю приходилось наблюдать, как в чьем-либо изложении самые сложные концепции вдруг становились ясными. Если мне удастся найти подходящие слова, чтобы донести некоторые ключевые идеи до других членов Flash-сообщества, я буду счастлив.

В этой книге я хочу поделиться с читателем своими мыслями и наиболее удачными разработками. Мы обсудим способы динамического создания изображений с помощью кода. Мы будем разрабатывать код на языке ActionScript, придерживаясь принципов объектно-ориентированного программирования. Однако эта книга, в первую очередь, описывает *процесс*. Я не хочу просто сообщить набор фактов о Flash или ActionScript. Мой особый подход к Flash основан на всем моем предыдущем опыте, и поэтому я постараюсь раскрыть перед читателем ход мыслей, приводящий к созданию кода.

Моя биография

Моей первой в жизни страстью была музыка. Мама начала учить меня игре на фортепьяно, когда мне было четыре года. По ее словам, первые несколько лет я занимался по два часа в день. Не знаю, почему это так увлекло меня, но мама говорит, что в детстве я на все отводил по два часа, будь то чтение, игра в Лего или музыка. Следующие 14 лет я совершенствовался в игре на фортепьяно и ежегодно участвовал в городских и областных конкурсах. В юности моей главной целью было стать концертирующим пианистом. Я играл на тромбоне и фаготе в школьном оркестре и пел в церковном хоре. После школы я увлекся игрой на акустической гитаре, кельтском свистке и джембе (африканский барабан).

Моей второй любовью было чтение. Все книжки, которые у меня были, я прочитал минимум дважды. Я выбирал какую-нибудь букву алфавита (*пусть сегодня будет... «К»*) и весь день читал статьи на эту букву во Всемирной книжной энциклопедии 1969 года издания. Я читал этикетки на бутылках с шампунем и инструкции ко всем электроприборам в доме. Дошло до того, что родители запрещали мне читать целый день.

Следующим моим большим открытием стали компьютеры. Когда мне было 10 лет, родители купили XT. Возможно, читатель помнит, что такое 486. Так вот, XT – это 086. Зеленый монохромный экран, 640 килобайт ОЗУ, 20-мегабайтный жесткий диск. (Один умный человек сказал: «Никому не нужно больше 640 К». Сейчас он владеет фирмой Microsoft.) Я изучал MS-DOS и, к своему восторгу, однажды обнаружил программу Basica – интерпретатор языка BASIC. Я купил книгу с программами на этом языке, якобы посвященную Олимпийским играм. Послушно набрав на клавиатуре огромный, как мне тогда казалось, объем кода, я был слегка разочарован тем, что увидел. Символы ASCII, прыгающие по экрану, лишь отдаленно напоминали плавание и забег на 100 м. Тем не менее дверь была открыта. Я продолжал исследовать BASIC и в конце концов научился переключать экран в различные графические режимы и рисовать точки и фигуры.

Я учился в старших классах, когда мои родители купили CorelDRAW! 2.0, и моей радости не было предела. К этому времени у нас уже был 286-й с частотой 12 МГц и одним мегабайтом памяти. На нем можно было запустить Windows 3.1, но она работала медленно. Работа в CorelDRAW! на таком компьютере требовала большого терпения; я, наверное, чаще видел песочные часы, чем курсор. Однако я изучил векторную графику, кривые Безье и познакомился с огромным количеством ужасных градиентов и изображений, которые в то время казались великолепными. Тогда же мои родители открыли новое дело – фирму«Rainbow Copy Center», которая первой в городе стала изготавливать цветные фотокопии. Я работал в ней по выходным и на летних каникулах и с помощью CorelDRAW! разрабатывал дизайн брошюр и рисунки на футболках.

Потом я увлекся математикой, впрочем, не сразу. В девятом классе она нагоняла на меня скуку, а учительница меня просто раздражала. В конце концов, я бросил делать домашние задания и стал валять дурака вместе с худшей половиной класса. Однажды мне взбрело в голову написать маленькими буквами множество зашифрованных сообщений, чем я и занялся на контрольной по математике. Вскоре меня вызвали в кабинет завуча. Она спросила, нет ли у меня каких-либо проблем. Я ответил, что нет, все в порядке. Тогда она спросила, почему в контрольной работе я написал: «О ГОРЕ МНЕ». Я рассмеялся и объяснил, что просто дурачился, потому что мне было скучно! Неожиданно для самого себя я спросил: «А можно перескочить по математике в следующий класс?» Завуч сказала, что подумает. Через некоторое время мне дали программу по математике за девятый класс, и я прошел ее к Рождеству. Через полтора года я экстерном практически завершил программу по математике за двенадцатый класс. (Страшно подумать, что было бы, если бы я продолжил заниматься математикой в девятом классе вместе со всеми.) В одиннадцатом классе я познакомился с физикой и вычислительной математикой и с тех пор дружу с ними. Я быстро научился *видеть* физику вокруг себя: силы, ускорение, гравитацию, трение – стоит только посмотреть.

Университетские годы

В старших классах я собирался стать либо программистом, либо профессором математики. Я поступил в университет, не сомневаясь в поставленной цели, и изучал математику и компьютерные дисциплины в течение двух лет. Я прошел пять математических курсов (методы вычислений I и II, вычисления со многими переменными, линейная алгебра и анализ комплексных функций), физику и три компьютерных курса. Но мне опять пришлось изменить специальность из-за того, что мне не нравился преподаватель. Меня так раздражали занятия по компьютерным дисциплинам, что я подумал: «Нет, нельзя потратить остаток жизни на программирование» и стал искать другую специализацию. Я переключился на философию и начал готовиться к карьере профессора философии.

Я уже был аналитиком по складу характера, но изучение философии удесятерило мои силы. Два из обязательных курсов были посвящены логике. Мы учились облекать в символическую форму логические (и нелогичные) аргументы. Пользуясь почти математическими формулами, мы доказывали справедливость (или ошибочность) умозаключений. Мы в совершенстве изучили работу таких логических конструкций, как И, НО, ИЛИ и ЕСЛИ...ТО. Помимо курсов логики расписание философского факультета было до предела заполнено абстрактными дисциплинами. Это был целый мир разнообразных многосложных «измов» (таких как *эпифеноменализм*), которые переворачивали наши представления о Вселенной.

Я научился организовывать идеи в мыслительные категории и находить интересные связи между ними. Как в хорошо структурированном XML-документе, мне стали ясны различные информационные узлы и их иерархическая организация. Я научился выделять самую суть идеи и по-разному доносить ее до разных людей. Поскольку философы безжалостны при анализе чужих высказываний, мои письменные работы стали более сжатыми и точными. Эта привычка пригодилась мне, когда я стал программировать на ActionScript. В следующих главах читатель заметит мою страсть к четкому, хорошо организованному и оптимизированному коду. Мне нравится объектно-ориентированное программирование (ООП); оно доставляет мне философское (и практическое) удовольствие. Со стороны может показаться, что изучение философии было излишним «крюком» на моем пути, но философия хорошо служит мне в других областях.

Однако перед получением степени бакалавра мой мозг был иссушен занятиями. Чтобы закончить университет вовремя, я должен был пройти 12 философских курсов за два года, а это действительно нелегко. Я уже ни в чем не был уверен – ни в физическом, ни в метафизическом. Желая заняться чем-то практическим, я снова оказался перед компьютером. Только на этот раз меня заинтересовал Интернет. Я хотел создавать веб-страницы и соединять их ссылками. А если бы мне удалось создать полосу меню с раскрывающимся подменю, то это было бы верхом совершенства (по крайней мере, я так тогда думал).

Курсы повышения квалификации

Так получилось, что через неделю после окончания университета я приступил к интенсивной девятимесячной образовательной программе на курсах повышения квалификации. Это учебное заведение в качестве цели декларировало «превращение выпускников университета в профессионалов в области информационных технологий», и можно с уверенностью сказать, что эта цель была достигнута. Мы изучали объектно-ориентированное программирование и проектирование, а также процесс быстрой разработки приложений. Занимались мы 5 дней в неделю по 5 часов в день (два часа в аудитории и три часа «работы в команде»).

Такой подход, включающий выработку навыков совместной работы, был уникальным аспектом программы. В командах по пять-шесть человек мы разработали четыре значительных приложения в таких принципиальных областях, как HTML, Visual Basic, Java и Oracle. Например, мы создали приложение под Windows и веб-сайт, которые позволяли вымышленному агентству по трудоустройству вести базу данных, включающую в себя соискателей, фирмы и вакансии. Поскольку каждый проект занимал два-три месяца, мы проходили весь процесс разработки приложения. Мы получали условия подряда от воображаемого клиента и должны были сообща разработать ответный документ и представить эскиз проекта. После этого нам передавали требования

клиента к проекту. Мы тщательно планировали архитектуру приложения и составляли проектную документацию на языке UML (к которому я вернусь в главе 2).

После курсов повышения квалификации я ничего не разрабатывал на Java, Visual Basic или в Oracle, но именно там я освоил объектно-ориентированную методологию, в принципе применимую ко многим платформам. Что еще важнее, я владел *процессом* – дисциплинами, упоминавшимися в этой главе, которые сформировали мою жизнь и карьеру.

Первая встреча с Flash

Кажется, впервые слово «Flash» я прочитал на сайте Matrix *www.whatisthematrix.com*. До того я уже просмотрел несколько сайтов с подключаемым модулем Shockwave. Как правило, они раздражали меня тем, что вначале необходимо было подключить модуль, а потом ждать (и ждать), пока сайт загрузится. Когда сайт Matrix затребовал модуль «Shockwave Flash», я, помнится, подумал: «Что за Flash? Ах да, *еще один* добавляемый модуль!» Сайт загружался целую вечность, но какой у него оказался интерфейс! Кругом сплошная анимация, все экраны содержат сложные раскрывающиеся меню – видеофильм, да и только. А звук... звуковые эффекты производили потрясающее впечатление на фоне тишины, царившей тогда в Сети.

Через несколько месяцев, уже на курсах повышения квалификации, я стал чаще заходить на Flash-сайты. Помню, как на сайте *Turtleshell.com* я впервые испытал эффект погружения. Казалось, я нахожусь в самом сайте и скольжу по нему от раздела к разделу. Мне понравился его спокойный чистый стиль, который повлиял на всю мою последующую работу. В конце 1999 года я посетил сайт Balthaser и чуть не свалился со стула. Трудно было поверить, что такое может происходить в моем броузере: это был полноэкранный видеофильм, только... еще лучше.

В то время некоторые мои соученики на курсах изучали Flash, и я однажды заглянул через плечо приятеля, когда он создавал фильм Flash 4. Я разглядел что-то вроде временной диаграммы – загадочное переплетение слоев, точек и горизонтальных линий с маленькими стрелками. Это выглядело пугающе, и я не стал вникать глубже. По крайней мере, до того «рокового» дня за два месяца до моего окончания курсов. Стремительно приближался последний экзамен по языку Java, но я устал учиться. Чтобы как-то отвлечься, я установил пробную версию Flash 4 и решил немного поиграть с ней. Через день-другой я прошел восемь уроков в меню «Справка», а также прочитал учебник в документации. Кроме того, я обнаружил несколько учебников на сайте Webmonkey. Три дня спустя я закончил свой первый Flash-проект: слайды для заключительной презентации моей команды по Java. Все другие команды воспользовались для этой цели приложением PowerPoint, и на этом фоне плавный твининг и сглаживание на Flash-слайдах смотрелись выигрышно.

Так я стал одержимым сторонником Flash. Последние два месяца учебы на курсах повышения квалификации я большую часть дневного времени либо изучал Flash, либо размышлял, что бы еще попробовать. На занятиях все студенты пользовались портативными компьютерами, и я непрерывно окрашивал и твининговал символы, пока преподаватель объяснял PL/SQL. В свободное время я разработал несколько небольших Flash-проектов. К счастью, это произошло в очень удачный момент с точки зрения учебы. Моих «текущих» навыков чуть-чуть не хватало, и мне пришлось подтянуться, чтобы использовать более сложные функциональные возможности Flash.

Ближе к окончанию курсов я испытывал противоречивые чувства. Мне нравилось программировать на Visual Basic и Java, но работать во Flash было просто божественно. Я мечтал немедленно начать карьеру Flash-разработчика. К моему сожалению, такие вакансии были редки, а у меня не было ни опыта, ни готовых наработок. Я примирился с необходимостью найти место программиста на какое-то время, а в нерабочие часы создавать задел для будущей карьеры.

Однако удача улыбнулась мне на последней неделе моего обучения на курсах. Приятель приятеля, работавший в маленькой фирме, которая занималась веб-разработками, пригласил меня на собеседование. Он оценил мой потенциал, и на следующий день после выпуска я приступил к работе в *monkeymedia.net*. Это было невероятно – получать деньги за удовольствие творить в среде Flash через два месяца после знакомства с ней. В течение следующего года я работал в *monkeymedia.net*, а затем по контракту и в Axis Interactive (*axis-media.com*). Все это время я быстро совершенствовал свои навыки. Казалось, что каждый месяц происходит что-то революционное, появляется новая концепция, которая кардинально меняет мой подход к работе во Flash.

Это была эпоха Flash 4, а анимация входила в мои основные обязанности. Художники фирмы разрабатывали компоновку, графику и раскадровку проектов. Я переносил эти составляющие во Flash и заставлял их петь и плясать. Я заполнял временную диаграмму твин-последовательностями для движения и изменения форм и сочинял соответствующую музыку и звуковые эффекты. Язык ActionScript имел тогда ограниченные возможности, так что и применял я его ограниченно, в основном, для создания кнопок, запускающих или останавливающих воспроизведение фильма.

Тогда же я приступил к разработке своего сайта-портфолио и завязал контакты во Flash-сообществе. В конце концов я стал находить клиентов самостоятельно, преимущественно в Соединенных Штатах. В феврале 2001 года я принял участие в конференции FlashForward в Сан-Франциско. Она произвела на меня сильное впечатление, о чем я еще расскажу в этой главе.

Flash – это феномен, который застал меня врасплох. Но он идеально вписался в мой опыт, словно вся моя предыдущая жизнь была подготов-

кой к роли разработчика мультимедийных проектов. Музыка, BASIC, CorelDRAW, математика, философия – все сошлось в фокальной точке. Мои прошлые занятия стали фундаментом моей новой страсти – Flash.

Дисциплина как центральное понятие

Я где-то слышал, что«успех – это момент, когда подготовка встречается с удачей». С одной стороны, мы достигаем успеха, только когда нам выпадает удача. В этом смысле чем больше у человека природных способностей, тем больше ему предоставляется удобных случаев их проявить. Впрочем, удача и способности *не гарантируют* успеха. Его второй необходимой составляющей является подготовка. Когда наступает поворотный момент, успешнее оказываются те, кто построил нужный фундамент. Такая подготовка называется дисциплиной.

Что такое дисциплина?

Слово«дисциплина» для многих имеет негативное значение и стоит в одном ряду с«контролем» и«наказанием». Об этом приходится сожалеть, ведь само слово происходит от латинского disciplina, что означает«учение», путь к зрелости, силе и успеху. Лично мне всегда были интересны *дисциплины* в том смысле, что существуют разные виды деятельности, каждая из которых является дисциплиной.

Даллас Уиллард (Dallas Willard) дает превосходное определение дисциплины в моей любимой книге«Spirit of the Disciplines». Он определяет дисциплину как деятельность человека, направленную на совершение того, что нельзя достичь прямым усилием. Идея понятна каждому. Например, я знаю, что не смогу пробежать марафон, если просто приду в назначенный день и стартую с остальными участниками. Однако если я начну с малого и предприму практические шаги, я в конце концов достигну такой физической формы, что марафон будет мне под силу.

Упомянутые практические шаги не сводятся к одному лишь бегу. Подготовка к марафону означает определенный стиль жизни, строгий режим работы и сна, диету, твердость воли. Основа успеха закладывается с помощью множества смежных дисциплин и состоит из многих кирпичиков, привычек.

Привычки

Нашей жизнью управляют привычки, как хорошие, так и дурные. Многие наши удачи и неудачи проистекают из привычек, выработанных давным-давно, как правило, в детстве. Так неужели мы обречены быть рабами наших плохих привычек, сформированных воспитанием? К счастью, нет. Мы, люди, обладаем уникальной способностью оценивать собственные мысли и меняться в определенной степени. Можно сказать, что наш мозг является особым типом аппаратно-про-

граммной системы, которая в состоянии перепрограммировать себя. И вот здесь на первый план выступают дисциплины.

Дисциплины – это сознательные попытки распознать плохие привычки в нашем образе жизни и заместить их хорошими. Очень трудно избавиться от плохой привычки, просто пытаясь не совершать соответствующие поступки. Сильные привычки непроизвольны и, так сказать, «автоматичны», поэтому они и являются привычками. Словечки, которые постоянно слетают с языка, важные события, о которых мы постоянно забываем, – все это происходит вопреки нашим лучшим намерениям. Привычки живут своей жизнью. Некоторые оказываются так сильны, что легче изменить почерк, чем отказаться от них. Иногда приходится предпринимать нетривиальные меры.

В университете мой образ жизни не имел структуры и не отличался последовательностью. Лекции были для меня, как флажки на пути слаломиста, и я маневрировал среди них, беспорядочно чередуя учебу, общение с другими студентами и сон. Большинство моих работ было написано в последнюю минуту, и сдавал я их с опозданием, несмотря на бессонные ночи. После нескольких лет такого сумасшествия я начал строить карьеру, работая на себя. Было невероятно трудно постигать законы бизнеса и зарабатывать деньги. Но труднее всего было воевать с плохими привычками, укоренившимися в моем стиле жизни.

В какой-то момент я оказался занятым в долгосрочном проекте, потребовавшем жесткого графика работы для постоянного продвижения вперед. Оказалось, что этого не так легко добиться. Я жил один и имел слишком много свободы и слишком мало самоконтроля. Мое рабочее время переплеталось с нерабочим, что не способствовало ни работе, ни отдыху. Я стал отставать от графика и впал в отчаяние.

Однажды, когда я рассказывал об этом нескольким близким друзьям, одному из них пришла в голову отличная идея. Он предложил мне купить дешевый портативный компьютер и решить, что в определенные часы я должен выходить из дома и «идти на работу». Мысль мне понравилась, и я предложил установить для меня ощутимое наказание за прогул, чтобы усилить мотивацию. Мы перебрали несколько интересных вариантов и остановились на следующем: всякий пропуск рабочих часов карался мытьем санузла у одного из моих друзей. Смешно, но это сработало. У меня вдруг появилась побудительная причина каждое утро вылезать из кровати и выходить на улицу. Моя ненависть к мыльной пене стала источником положительной энергии, а мое отвращение к унитазам – стимулом к обновлению. В конце концов мне пришлось вытерпеть последствия прогула, но лишь несколько раз. Через некоторое время такой рабочий график стал мне привычен, он внес в мою жизнь элемент упорядоченности, помогающий структурировать остальную часть дня.

Без прочной основы, построенной в течение определенного времени с помощью дисциплины, мы не сможем достичь наших целей несмот-

ря ни на какие благие намерения. Природные способности играют заметную роль в наших достижениях, но дисциплина всегда необходима для целостного и долгосрочного успеха.

Мои дисциплины

Я хотел бы назвать ряд дисциплин, оказавшихся особенно полезными для меня как Flash-энтузиаста. Успех в любом виде деятельности основан на множестве принципов и привычек. Некоторые очевидны, например профессиональные навыки, честность, упорство в стремлении к цели, пунктуальность. Эти качества легко перечислить, но овладеть ими гораздо сложнее. Мне работа над собой в этих областях дается с большим трудом. Но есть и другие личные дисциплины, глубоко укоренившиеся в моей жизни. Мне хочется рассказать читателю о том, как они помогли мне стать специалистом по Flash.

Самообразование

По характеру я человек независимый. Когда дело касается учебы, я глубоко убежден, что сам несу ответственность за свое образование. Если я хочу что-то узнать, я ищу информацию в справочнике. Если читая книгу или работая над статьей, я встречаю незнакомое слово, я тут же смотрю в словарь. Так и во Flash, когда я сталкиваюсь с чем-то непонятным, я пытаюсь разобраться самостоятельно. Когда я еще только изучал Flash (версию 4.0), я при *каждой* проблеме читал электронную справку. За шесть месяцев я, должно быть, изучил 95 процентов документации.

Эта привычка так сильна, что я едва ли ощущаю ее как дисциплину. Мне не пришлось вырабатывать ее; она просто составляет часть моей жизни. Я рос с убеждением, что в большинстве случаев я почти без усилий *смогу* выяснить все, что мне потребуется. И я прибегаю к чужой помощи лишь после того, как исчерпаю все возможности самостоятельно. Конечно, все люди разные, и у каждого свой стиль приобретения знаний. Некоторым легче учиться в коллективе; у одних лучше развита вербальная память, у других – механическая. Моя жизнь не может служить стандартом для окружающих. Я не могу, даже с натяжкой, утверждать, что чтение энциклопедии или словаря просто для удовольствия является «нормальным делом», но я все-таки хочу сказать, что изучение справочных материалов принесло мне огромную пользу. Оно позволило мне учиться быстро и пользоваться многими инструментами более эффективно.

Один мой университетский приятель открыл собственную фирму по дизайну печатной продукции и веб-дизайну сразу после получения диплома биолога. Он рассказывал мне, что прочитал руководство Macromedia Dreamweaver 2 за два дня. Немного спустя он прочитал руководство Flash 4 за один присест. Сегодня он руководит успешной дизайнерской компанией.

Должен признать, что его методы самообразования весьма интенсивны и не каждому подходят. Лично мне не удавалось прочитать все руководство так быстро. Важно другое. Руководство для того и существует, чтобы его читали. Это требует целеустремленности, но всегда окупается. В январе 2001 года я решил, что пора переходить на Flash 5 и изучать новый язык ActionScript. Я читал руководство по ActionScript, сидя на диване, по часу в день, как художественную литературу. Эта дисциплина, как и некоторые другие мои привычки (см. далее), помогла мне довольно быстро освоить Flash 5.

Тому, кто не очень любит изучать руководства и справочники, я посоветую попробовать это в качестве дисциплины. Разнообразная информация по тысячам устройств и программ ждет своего читателя на книжных полках. Аббревиатура R.T.F.M. (Read The «Friendly» Manual, прочитайте «дружественное» руководство[1]) – это не приговор, а приглашение к независимости и росту.

Специализированная практика

Когда я играл на фортепьяно, я обычно занимался меньше, чем следовало. В старших классах я поставил себе целью ежедневные часовые уроки музыки, но не сумел добиться регулярности. Впрочем, я научился заниматься «эффективно». Со временем я выработал различные стратегии и подходы к разучиванию музыкальных произведений. Я сосредоточивался на ключевых моментах и прибегал ко всяческим ухищрениям, позволяющим овладеть ими и превратить мелодию из упрямого мула в скаковую лошадь.

Цель практики в том, чтобы играть произведение правильно, но достижение этой цели требует огромного количества *неправильных* попыток. Я намеренно играл быстрые пассажи медленно, тихие ноты – громко, плавную мелодию – отрывисто и свинговал жесткий ритм. И вдобавок ко всему я повторял это снова и снова. (Мне жаль моих родителей, которые были вынуждены выносить эту сумасшедшую какофонию. В конце концов они поменяли потолок и установили скользящую стеклянную дверь, чтобы хоть как-то уменьшить шум из подвала, где я репетировал.) Нередко я брался за музыкальные произведения более высокого уровня; преодоление технических сложностей помогало мне справиться с обычными произведениями. Все эти приемы позволяли мне «выжать максимум» из моих недолгих занятий. Я так быстро совершенствовал свои навыки, что к концу школы закончил программу музыкального образования по классу фортепьяно. Однажды я завоевал второе место на областном музыкальном конкурсе, и сегодня мой репертуар состоит из большого количества красивых и технически сложных музыкальных произведений.

---

[1] Расшифровка этой аббревиатуры, принятая на просторах Интернета, слишком груба для печатного издания. Видимо, поэтому автор написал слово «дружественное» в кавычках. – *Примеч. ред.*

Когда я стал изучать Flash, то быстро осознал необходимость специализированной практики. Приобретя любой навык, я видел, что нужно овладеть еще двумя. Я стал записывать, какие технические приемы мне следует освоить, по мере того как сталкивался с ними. Так получился список, который во времена Flash 4 выглядел примерно следующим образом:

Что изучить во Flash

- Всплывающее меню
- Перетаскивание видеоклипа
- Загрузка фильма
- Загрузка переменных из текстового файла
- Установка свойств видеоклипа

И так далее...

Шаг за шагом я проходил по этому списку. Я изучал различные концепции (конечно, по руководству пользователя) и проверял их на практике либо в проектах, либо в небольших тестовых фильмах.

Важно отдавать себе отчет в том, что именно нам неизвестно. Помню, как однажды я натолкнулся на статью о Flash 4 под названием «Умеете ли вы применять Load Movie?» Я сразу подумал: «Load Movie? Что это такое? Похоже, что не умею». В статье говорилось, что загрузка внешних SWF-файлов является важнейшим умением, которое влияет на всю деятельность Flash-разработчика. Я поверил автору на слово и решил изучить команду Load Movie. До того как я прочитал статью, я находился в неведении относительно своего неведения. У меня и в мыслях не было, насколько эта техника важна для моей карьеры в области Flash. Автор был прав – по мере того как я практиковался в использовании Load Movie, структура моих проектов менялась революционным образом.

На каком бы этапе пути вы ни находились, проведите перепись белых пятен в своем знании. Проанализируйте свой стиль изучения материала, поищите приемы, которые позволили бы продвигаться быстрее, составьте для себя расписание практических занятий.

Пример: изучение «горячих» клавиш

Большинство команд Flash доступны через систему меню. Однако многим из них соответствуют комбинации «горячих» клавиш. Я с самого начала решил изучить как можно больше этих комбинаций и пользоваться ими. Этот процесс может послужить примером применения таких дисциплин, как самообразование и специализированная практика. Я чувствовал, что комбинации «горячих» клавиш являются необходимым элементом «овладения основами», и разработал для этого некоторые практические процедуры.

В табл. 1.1 приведены мои основные комбинации «горячих» клавиш. Сейчас я пользуюсь ими постоянно, даже не задумываясь.

Примечание

Здесь я не указал много других «горячих» клавиш для операций, не специфичных для Flash, таких как вырезка, копирование, отмена, переключение и закрытие окон. Кроме того, я перечислил только клавиши для Windows. Я уверен, что пользователи Macintosh сообразят, как перенести их в свою систему.

Мне пришлось приложить определеннные усилия, чтобы запомнить эти комбинации. Я заставил себя пользоваться комбинациями «горячих» клавиш, создав новую дисциплину: когда мне была нужна какая-нибудь команда меню, я запоминал ее клавиатурный эквивалент. Затем я выходил из меню и вызывал команду с клавиатуры. Я понимал, что до следующего применения этой команды я, вероятнее всего, забуду комбинацию клавиш и снова открою меню. Такая практика постоянного возврата к меню поначалу замедляла мою работу, но я был вынужден учиться быстро. Сейчас я практически не пользуюсь ни главным, ни контекстным меню.

Таблица 1.1. Мои основные «горячие» клавиши во Flash

Комбинация клавиш	Описание
F8	Преобразовать в символ
Ctrl-F8	Новый символ
F5	Вставить кадр
Shift-F5	Удалить кадры
F6	Вставить ключевой кадр
Shift-F6	Стереть ключевые кадры
F7	Вставить пустой ключевой кадр
Ctrl-Alt-X	Вырезать кадры
Ctrl-Alt-C	Скопировать кадры
Ctrl-Alt-V	Вставить кадры
Ctrl-Shift-V	Вставить вместо (одна из моих любимых команд)
Ctrl-Enter	Протестировать фильм
Ctrl-F12	Опубликовать предварительную версию (F12 во Flash 5)
Shift-F12	Опубликовать

Почему я столько внимания уделяю комбинациям «горячих» клавиш? Неужели они так важны для успешной карьеры Flash-разработчика? По правде говоря, нет. Я уверен, что можно прекрасно обойтись и без SHIFT-F5 или CTRL-F8. В этом примере в первую очередь важен

описанный процесс. Я хотел научиться быстро работать во Flash. Я предвидел, каким образом знание «горячих» клавиш поможет мне достичь цели, и действовал соответственно. Намеренно нарушая обычную рабочую процедуру, я сумел на ранней стадии выработать хорошие привычки, которые приносят мне пользу по сей день. Комбинации «горячих» клавиш сэкономили мне массу рабочего времени и вообще сделали процесс разработки более гладким. Это небольшой, но конкретный пример важности дисциплины.

Сообщество

Вокруг всякой дисциплины, искусства или ремесла обычно формируется некое сообщество. Вначале ремесло дает толчок к развитию сообщества, затем сообщество поддерживает ремесло. Трудно работать в полной изоляции. Совместная работа равноправных сотрудников создает интересное сочетание духа товарищества и конкуренции, что составляет важную часть мотивации.

Когда я занимался игрой на фортепьяно, весь мой год строился с прицелом на музыкальный конкурс, который происходил весной. Каждый год я разучивал новые музыкальные произведения и думал только о тех кратких мгновениях, когда я представлю публике результат своего труда и получу аплодисменты зрителей и критику жюри. Возможно, я слишком сильно сосредоточивался на соревновательном аспекте музыкального конкурса. Безусловно, раскрыть красоту ноктюрна Шопена гораздо важнее, чем «опередить Бева и Женевьеву в этом году». Тем не менее именно чистая дружба с другими юными музыкантами, нашими учителями и родителями поддерживала и стимулировала меня.

Сообщество Flash-пользователей – явление весьма интересное. Поражает интенсивность, с которой множество людей со всех концов света объединилось вокруг одного программного продукта. Впрочем, это больше, чем просто приложение. Flash превращается в стиль жизни; переполняет голову идеями, не давая заснуть; отвлекает от повседневной работы («держу пари, то же самое можно сделать во Flash»).

Когда я застревал в колее шаблонных решений, сообщество подсказывало мне новые идеи. Когда мной овладевало искушение почивать на лаврах, находились люди во Flash-сообществе, поражавшие меня фантастической изобретательностью. В моей жизни появились две специфических дисциплины, связанные с сообществом. Это *изучение во время преподавания* и *открытый исходный код*.

Изучать, преподавая

Хорошо известно, что преподавание – лучший способ глубже понять тему. В старших классах я время от времени работал репетитором по математике и информатике и в течение двух лет давал уроки игры на фортепьяно. В университете я был ассистентом преподавателя компь-

ютерной технологии, проверял студенческие работы по программированию и консультировал однокурсников.

Как-то раз я объяснял одной своей знакомой концепции, которые изучал на факультете философии. Помню, как она сказала: «Роберт, ты хороший студент».

Я запротестовал: «Что ты хочешь сказать? Я сплю на лекциях, пропускаю семинары, постоянно откладываю все дела на потом, с опозданием сдаю учебные работы».

«Я не это имею в виду, – ответила она. – Ты сердцем принимаешь то, чему учишься». Должен признать, она была права. Я горячо интересовался всем, чему меня учили, и постоянно рассказывал об этом окружающим. Например, я был сильно увлечен экзистенциализмом и все уши прожужжал друзьям на эту тему. Но поскольку при этом я не мог употреблять непонятные профессиональные термины, мне приходилось излагать мысли простыми словами. Мне было радостно видеть, что собеседники воспринимают философские идеи, и я глубже познавал предмет, уча других.

Когда мои отношения с Flash находились в фазе изучения, я все время «рылся» в руководстве пользователя и электронном справочнике. Однако самые глубокие знания я получил, в основном, отвечая на вопросы в форумах. Я знал, что существует масса тонкостей, в которых я еще не разобрался. Для меня мотивацией в исследовании новых областей была необходимость найти информацию, нужную другим людям.

Например, кто-нибудь спрашивает: «Как удалить элемент массива?» Я говорю себе: «Хороший вопрос. А как бы я удалил элемент массива?» Я ищу ответ в руководстве по Flash, а если не нахожу его, то ищу в Интернете. Затем я пишу код, тестирую его во Flash и отправляю ответ в форум. Это действительно отличная система. Вы сразу убиваете двух зайцев – помогаете другому и учитесь сами.

Открытый исходный код

Поворотный пункт в моем общении с Flash наступил в феврале 2001 года, и связан он был с понятием «открытый исходный код». Сайт, который я разработал по контракту с Axis Interactive (*axis-media.com*), был дважды номинирован на фестивале Flash-фильмов. Поэтому фирма Axis отправила меня со своими представителями в Сан-Франциско для участия в фестивале и конференции FlashForward.

В то время я был принципиальным сторонником «закрытого исходного кода». Я считал, что FLA-файлы следует хранить в секрете, чтобы защитить свои оригинальные идеи и оставаться конкурентоспособным. К тому же, незадолго до описываемых событий меня обокрали, сорвав важный для меня эксперимент. Воры «вынесли» код прямо из SWF при помощи ActionScript Viewer, а затем выдавали за свой, предлагая FLA-файлы для загрузки. Я смог доказать свое авторство, но остался подозрительным и осторожным.

Однако в Сан-Франциско я посетил семинар Джошуа Дэвиса (Joshua Davis) и услышал его мнение об открытом исходном коде. Меня вдохновили его щедрость, увлеченность и взгляд на жизнь. Я был свидетелем прекрасного парадокса: чем больше мы отдаем, тем больше мы имеем. Я понял, что открыв свою работу, я получу больше, чем потеряю. Во всяком случае я уже убедился, что SWF-файлы защитить практически невозможно. Вернувшись с конференции, я решил раскрыть свои FLA-файлы.

Следуя призывам Самуэля Вэна (Samuel Wan) создавать ясный код, я немедленно переписал все, что создал на ActionScript, и добавил многочисленные комментарии. Объявив о доступности своих работ, я обнаружил, что трафик на моем сайте взмыл под облака (с 5 посетителей в день до 100). Оказалось, что опубликование кода является практической защитой от воров. Чем больше людей осведомлены о моей работе, тем труднее кому-нибудь выдать ее за свою.

Раскрытие своего исходного кода – отличный способ довести его до совершенства. Следует лишь учесть предложения по его улучшению. Необходимость предстать перед публикой стимулирует более тщательную проработку идей и создание некоего личного стиля. Это способ внести вклад в копилку Flash-сообщества, а в ответ получить признание за уникальные идеи, способные заинтересовать разработчиков в большей степени, чем клиентов.

У английских моряков есть поговорка: «Прилив поднимает все корабли». Я верю, что успех Flash во многом обеспечен той щедростью, с которой делятся идеями члены сообщества. Конечно, находится место и для *коммерческих* Flash-ресурсов, и они приобретают все возрастающую важность по мере того, как Flash развивается как платформа для разработчиков. Многие люди тратят немало времени на создание качественных инструментальных средств для Flash, в частности компонентов, и я буду рад, если они получат материальное вознаграждение за свою работу.

Итеративный процесс

На курсах повышения квалификации мне неоднократно приходилось слышать фразу: «Это итеративный процесс». Она даже стала дежурной шуткой. Когда приложение, которое мы разрабатывали командой, давало очередной сбой, мы переглядывались и острили: «Ну, это итеративный процесс». Итерация – это просто очередное повторение цикла. Идея итеративного процесса заключается в том, что проект разрабатывается поэтапно, за несколько проходов по кругу. Иногда самым легким путем из пункта A в пункт B является не прямая, а несколько витков спирали (например, лестница в доме).

Представьте себе, как растет дерево. Не бывает так, чтобы сперва вырастал ствол, затем одна ветка, другая и так далее. Дерево начинается

с ростка, так сказать, с «прототипа». Молодое деревце во многом похоже на взрослое, но его размер меньше и детали структуры выражены в меньшей степени. Дерево проходит через ряд циклов, строя себя и разветвляясь в ритме времен года. Годичные кольца являются записью этого итеративного процесса.

Многие вещи в нашем мире естественным образом происходят в цикле. Они начинаются с элементарных форм и постепенно наращивают свою функциональность и сложность. Очевидным примером являются живые организмы, но то же самое происходит и с организациями. Например, общественный/ресурсный веб-сайт поначалу невелик. Однако по мере появления новых членов сообщества он приобретает больше функциональных особенностей, которые привлекают все больше людей, и так далее.

Развитие по спирали

Я обнаружил, что обучение не является линейным процессом. Знание – это не вереница отдельных фактов, идущих по конвейеру и поглощаемых друг за другом. Оно больше похоже на паутину переплетающихся идей. Это своеобразный Интернет в голове, причем постоянно развивающийся. Знание расширяется циклически и во многих направлениях. Наше первоначальное представление о новой концепции эскизно. Оно слабо связано с другими, более знакомыми концепциями. Однако знание разрастается по мере того, как мы возвращаемся к прежним мыслям, формируем новые связи и прорабатываем контекст.

Я нередко сталкиваюсь с кодом или идеями, находящимися за пределами моих знаний; особенно часто они встречаются в списке рассылки Flashcoders. Когда объем или сложность материала превышает определенный уровень, я не пытаюсь понять его целиком. Я просто пробегаю глазами текст и стараюсь выявить основные положения. Если тема кажется мне важной, я откладываю текст на некоторое время. Иногда я бываю просто не готов воспринять новую идею, – у меня нет теоретического каркаса для ее поддержки. Но тем не менее я остаюсь открытым для нее и создаю в уме еще нечеткий узел, который в конце концов может превратиться в нечто полезное.

Помню, весной 2001 года во Flashcoders развернулась дискуссия относительно слушателей и обработчиков событий, которая как-то «проплыла» надо мной. Впрочем, я сохранил все объемистые послания и перечитывал их в течение следующих недель. В конце концов у меня «щелкнуло» в голове, и теперь слушатели являются одной из моих любимых концепций.

Те, кто изучает язык ActionScript, могут придерживаться аналогичного подхода. Читайте ActionScript Dictionary за несколько проходов, каждый раз погружаясь все глубже. При первом проходе обращайте внимание на базовые структуры языка: операции, условные операторы, циклы, объявления функций и т. д. Второй проход пусть будет об-

зором встроенных классов языка ActionScript: `Array`, `Button`, `Color`, `MovieClip`, `TextField` и др. Представлять, какие инструменты находятся в вашем распоряжении, очень полезно. Следующий этап может включать в себя просмотр *методов*, которыми обладают объекты. Основной принцип состоит в том, чтобы начать писать картину широкими мазками, а затем прорабатывать детали.

Расстановка приоритетов

На курсах повышения квалификации нам особенно настойчиво внушали идею об относительной важности «требуемых функциональных возможностей» по сравнению с «дополнительными». У каждого проекта были подробные спецификации, перечисляющие основные функциональные возможности, которые было необходимо реализовать, и это была «требуемая функциональность». Например, один проект заключался в разработке приложения для службы занятости, которое вело бы учет фирм, вакансий и претендентов на рабочие места. Требования клиента включали в себя процедуры добавления фирм, вакансий и претендентов в списки, редактирования сведений о них и автоматического поиска вакансий для претендентов. Если приложение не будет предоставлять пользователю хотя бы одну из требуемых возможностей, разработчики будут строго наказаны.

Кроме того, нам предоставлялся список «дополнительных возможностей», которые клиент приветствовал бы, если бы у нас нашлось время на их реализацию. За них можно было получить премию, которая в нашем случае выражалась в оценках. Однако если бы хоть одна из *требуемых* возможностей отсутствовала, ни о какой премии не могло быть и речи. Такая постановка задачи учила нас расставлять приоритеты. В большинстве проектов удавалось реализовать некоторые дополнительные возможности, но все силы мы в первую очередь бросали на разработку требуемых возможностей.

Этот принцип можно сформулировать так: излишнее не стоит и гроша при отсутствии необходимого. Клиенту нравится глазурь на булочке, но он предпочтет простую булочку глазури на блюдце. Для меня это было хорошим уроком. Оказалось, я «зацикливался» на деталях, добиваясь совершенства в одном направлении и пренебрегая другими. Со временем я научился периодически останавливаться и проверять, соответствуют ли действительности мои приоритеты и успехи.

Расстановка приоритетов особенно важна в вечно спешащем мире Интернета и веб-разработок. Когда временной ресурс ограничен, приходится выбирать, какую функциональность обеспечить в текущей версии проекта. Так мы возвращаемся к понятию итеративного процесса: приступая к проекту, следует относиться к нему как к версии 1.0. Первая версия не обязана обладать всей мыслимой функциональностью. Это отправная точка, фундамент для будущего здания.

Учет перспективы

Еще одним принципом итеративного процесса является *разработка с учетом перспективы*. Многие проекты проходят несколько этапов. Даже когда кажется, что работа закончена, может в какой-то момент возникнуть необходимость в модификации или расширении.

Например, я недавно узнал, что университет, где я учился, планирует строительство нового университетского центра с более просторным спортивным залом. Здание будет одним из самых больших в университетском городке, и стоить оно будет много миллионов долларов. Чтобы облегчить бремя расходов, центр будет строиться в несколько этапов. Вначале будет возведена основная часть здания, которую можно эксплуатировать, пока сооружаются пристройки. Такой подход потребовал дополнительных затрат на проектирование, поскольку было необходимо спланировать поэтапное расширение центра. Однако эта инвестиция принесет университету большую выгоду в будущем.

Необходимость разработки с учетом перспективы диктуется здравым смыслом. Все понимают, что красить пол надо так, чтобы не оказаться потом в углу комнаты. Тем не менее многие следуют этому принципу не так часто, как надо, особенно при разработке кода для Flash. Из нескольких видеоклипов, цепочек кадров и команд совсем нетрудно соорудить нечто работающее. Плохо спланированный клип может хорошо выглядеть снаружи, но его «внутренность» будет представлять собой сочетание непонятной структуры и немодульного кода с жестко «зашитыми» значениями. При попытке использовать его в другом проекте или хотя бы настроить на другие пользовательские предпочтения вся конструкция рухнет, как карточный домик.

Мне попадалось немало программ для Flash, которые выглядели практичными, но стоило открыть FLA-файл, как меня передергивало. Как правило, это несколько десятков строк кода, небрежно брошенных в обработчик событий `onClipEvent`, без инкапсулирующих функций и с массой абсолютных, нединамических ссылок на `root` и другие переменные. Принципы объектно-ориентированного программирования (ООП) не соблюдены, и это пагубно влияет на читаемость кода и возможность его повторного использования. Мне жаль того, кто попытается разобраться в подобной мешанине и захочет заставить ее работать в другом проекте. Flash-программисты, я умоляю вас, выучите ООП и пишите код с учетом перспективы!

Я страстно ненавижу приземленные и рутинные задачи. Парадокс, но нередко я тружусь более усердно, когда стараюсь избавить себя от работы в будущем. Как-то во время летних каникул я получил от родителей задание сложить пополам и собрать в стопки несколько тысяч рекламных листков. Было очевидно, что на это уйдет много часов, а я хотел закончить как можно быстрее. Прежде чем приступить к делу, я некоторое время искал самый эффективный способ справиться с зада-