HIGHTECH

ДЖОИ ЛОТТ и др.

ADOBE AIR

Практическое руководство по среде для настольных приложений Flash и Flex

Символ®

Adobe AIR
in Action

Joey Lott, Kathryn Rotondo
Sam Ahn, Ashley Atkins

MANNING

HIGH TECH

Adobe AIR

Практическое руководство по среде для настольных приложений Flash и Flex

Джои Лотт, Кэтрин Ротондо,
Сэмюел Ан, Эшли Аткинс

Санкт-Петербург — Москва
2009

Серия «High tech»

Джои Лотт, Кэтрин Ротондо,
Сэмюел Ан, Эшли Аткинс

Adobe AIR. Практическое руководство по среде для настольных приложений Flash и Flex

Перевод С. Маккавеева

Главный редактор	*А. Галунов*
Зав. редакцией	*Н. Макарова*
Выпускающий редактор	*П. Щеголев*
Научный редактор	*М. Антипин*
Редактор	*Ю. Бочина*
Корректор	*С. Минин*
Верстка	*Д. Орлова*

Лотт Дж., Ротондо К., Ан С., Аткинс Э.
Adobe AIR. Практическое руководство по среде для настольных приложений Flash и Flex. – Пер. с англ. – СПб.: Символ-Плюс, 2009. – 352 с., ил.
ISBN-10: 5-93286-136-3
ISBN-13: 978-5-93286-136-3

Adobe AIR – кросс-платформенная среда исполнения для развертывания приложений Flash и Flex в качестве настольных или работающих в смешанном сетевом/автономном режиме. Приложения AIR устанавливаются и выполняются локально, поэтому у них есть доступ к файловой системе, что дает им преимущества над веб-приложениями.

Авторы начинают с простых вещей, знакомят с функциями AIR API, а затем показывают, как на практике создаются приложения AIR. Рассматриваются создание и настройка системных окон, а также обмен данными с локальной файловой системой или базой данных. Обсуждается, как AIR подключается к веб-сервисам и как устраняет разрыв между Flex и JavaScript. Книга хорошо иллюстрирована и содержит массу исходного кода, доступного для загрузки из Интернета. Издание предназначено для разработчиков, знакомых с Flash и Flex и стремящихся перейти от броузерных приложений к настольным.

ISBN-10: 5-93286-136-3
ISBN-13: 978-5-93286-136-3
ISBN: 1-933988-48-7 (англ)

Издательство «Символ-Плюс». 199034, Санкт-Петербург, 16 линия, 7,
тел. (812) 324-5353, www.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 30.10.2008. Формат 70x100 1/16 . Печать офсетная.
Объем 22 печ. л. Тираж 1500 экз. Заказ №
Отпечатано с готовых диапозитивов в ГУП «Типография «Наука»
199034, Санкт-Петербург, 9 линия, 12.

Оглавление

Предисловие

Мой друг Пол Ньюмен (да, это его настоящее имя, и нет, это не *тот* Пол Ньюмен) позвонил мне год назад и поинтересовался, не приму ли я участие в написании книги об Apollo (кодовое название Adobe AIR на тот момент). Я был сильно загружен и поэтому согласился лишь после некоторых колебаний. Я имел общее представление об Apollo, но лишь с этого момента я всерьез обратился к этой технологии. Позднее Полу пришлось оставить этот проект ввиду других обязательств, но я продолжал изучать Apollo и готовиться к написанию этой книги.

Ранее у меня существовал ряд предубеждений против Apollo. Я имел десятилетний опыт работы с Flash и Flex и широко пользовался средствами Flash и Flex для создания настольных приложений. Я делал во Flash самостоятельные исполняемые модули практически с момента начала работы с ним. Для встраивания дополнительных функций в настольные приложения, сделанные на Flash, я с разной степенью успешности использовал программы FlashJester, Northcode SWF Studio и Multidmedia Zinc и рассматривал Apollo как еще одну альтернативу этим программам. Честно говоря, меня даже несколько возмущало, что такая огромная корпорация, как Adobe, пытается сокрушить работающие компании, выпустив конкурирующий продукт. Однако, поработав с Apollo, я пришел к выводу, что этот продукт существенно отличается от всех других.

Вскоре Adobe заменила название Apollo на Adobe AIR. AIR дает возможность разработчикам использовать свои навыки программирования на Flash и Flex для создания настольных приложений. В этом отношении он очень похож на упомянутые мной продукты. Однако AIR не создает выполняемые модули для конкретной системы. Приложения AIR требуют исполнительной среды AIR. В этом отношении AIR похож не на такие программы, как Zinc, а на среду выполнения типа Java или .NET. Понимание этого изменило мой подход к AIR.

По прошествии почти года рукопись написана, отредактирована и подготовлена для печати. За это время мы, авторы, много узнали об AIR и постарались поделиться своими знаниями на страницах этой книги. Мы искренне надеемся, что издание окажется полезным читателям и поможет им лучше разобраться в том, как работать с AIR.

Джои Лотт

Об авторах

Джои Лотт (Joey Lott) обладает богатым профессиональным опытом работы с такими технологиями Adobe, как Flex, Flash и ActionScript. Он автор или соавтор «ActionScript Cookbook», «Programming Flash Communication Server», «The Flash 8 Cookbook» и ряда других книг в этой области. Вместе с Сэмом Аном он является компаньоном и основателем «The Morphic Group».

Кэтрин Ротондо (Kathryn Rotondo) – программист в Schematic. Она получила диплом программиста в Harvard Extension School и сертификат по Flash в родайлендской Школе дизайна.

Сэм Ан (Sam Ahn) занимался конструированием и созданием RIA на протяжении нескольких лет, в том числе для таких клиентов, как Pfizer, Wyeth, MINIUSA и Puma. Вместе с Джои Лоттом он является компаньоном и основателем «The Morphic Group», компании по разработке интерактивных приложений, занимающейся в основном приложениями Flash/Flex.

Эшли Аткинс (Ashley Atkins) – старший программист в «Six Red Marbles» с более чем шестилетним опытом разработки на ActionScript. Он занимался как разработкой простых интерактивных обучающих программ, так и проектированием и созданием приложений на Flex и AIR.

Благодарности

Авторы книги хотели бы поблагодарить всех, кто участвовал в выпуске печатного издания, которое вы держите в руках. Это большой список сотрудников Manning, которые помогли в создании книги. Майкл Стивенс (Michael Stephens) первым увидел потенциал издания Adobe AIR, за что мы ему очень признательны. Нермина Миллер (Nermina Miller) была редактором-консультантом этой книги; без нее сроки и качество издания, несомненно, сильно пострадали бы. Это благодаря Нермине у вас в руках находится законченный и связный текст! Мы хотим также сказать спасибо Бенджамину Бергу (Benjamin Berg) за редактирование окончательного варианта рукописи.

Мы выражаем свою благодарность Карен Тегтмейер (Karen Tegtmeyer) за организацию рецензирования рукописи, которое помогло нам улучшить ее содержание и ответить на комментарии первых читателей и коллег-рецензентов. Наличие обратной связи во время написания и редактирования книги было очень ценным для нас. Особой оценки заслуживают наши коллеги-рецензенты Бернард Фаррелл (Bernard Farrell), Райан Стьюарт (Ryan Stewart), Дасти Джуетт (Dusty Jewett), Кристофер Хаупт (Christopher Haupt), Тим О'Хара (Tim O'Hare), Роби Сен (Robi Sen), Майк Клаймер (Mike Clymer), Шон Мур (Sean Moore), Клинт Тредуэй (Clint Tredway), Джереми Андерсон (Jeremy Anderson), Пэтрик Пик (Patrick Peak), Оливер Голдман (Oliver Goldman), Джек Д. Херрингтон (Jack D. Herrington), Натан Левеск (Nathan Levesque), Бруно Лоуаги (Bruno Lowagie), Дэниел Тодд (Daniel Todd) и Брендан Мюррей (Brendan Murray).

Роберт Гловер (Robert Glover) был техническим редактором этой книги, а значит, прочел каждую главу и каждую строчку кода, чтобы гарантировать их точность с технической точки зрения. Нельзя не отметить важность этого этапа, и мы искренне благодарны Роберту за его прекрасную работу. Теплых слов заслуживает также выпускающая команда, работавшая над дизайном книги и обложки и выполнившая набор текста и корректуру: Дотти Марсико (Dottie Marsico), Тиффани Тейлор (Tiffany Taylor), Анна Уэллес (Anna Welles), Лесли Хеймс (Leslie Haimes), Гэбриэл Добреску (Gabriel Dobrescu), Рон Томич (Ron Tomich) и Мэри Пиргис (Mary Piergies).

Хотим также отдать дань уважения команде разработчиков AIR в Adobe за создание замечательного продукта, предоставление отлич-

ной документации и ответы на наши вопросы. Особое спасибо Оливеру Голдману, который не только подробно прорецензировал несколько глав, но и нашел время, чтобы лично ответить на электронные письма, касающиеся технических деталей AIR и цифровых подписей.

Без вас, наших читателей и коллег, не было бы необходимости и потребности в этой книге, поэтому мы хотим поблагодарить и вас за проявленные интерес и увлечение.

Об этой книге

Это книга об Adobe AIR для разработчиков на Flash и Flex. Хотя AIR-приложения можно создавать с помощью HTML и JavaScript, но данная книга посвящена исключительно применению Flash и Flex для создания AIR-приложений. Программные интерфейсы AIR для ActionScript и JavaScript весьма схожи между собой. Однако мы пришли к выводу, что попытка одновременно рассказывать о нюансах как JavaScript, так и Flash и Flex, приведет к бесформенности книги. Поэтому мы решили сосредоточиться только на Flash и Flex.

Возможно, некоторым читателям покажется, что мы и так попытались охватить слишком многое, объединив под одной обложкой Flash и Flex. Такая критика справедлива. Иногда нам приходилось идти на компромисс, включая в одних местах примеры кода, более подходящие для Flash, а в других – более подходящие для Flex. Но это можно расценивать не только как недостаток, но и как достоинство. Нам кажется, что в результате создан более широкий контекст для понимания AIR, что позволяет принимать более правильные решения при создании приложений AIR. В конечном счете вы сами решите, оправдан или нет наш подход, хотя, конечно, мы советуем рассматривать книгу с нашей точки зрения.

Кому адресована эта книга

Естественно, что эта книга рассчитана на разработчиков Flash и Flex, которые хотят воспользоваться своими знаниями при создании AIR-приложений. Разработка на Flash, ActionScript и Flex подробно описывается в десятках, если не сотнях книг. В данной книге мы не предлагаем никакого вводного материала на эту тему. Поэтому, если вы не знакомы с API Flash, Flex или стандартного ActionScript, эта книга может вызвать у вас затруднения. Мы рекомендуем сначала изучить основы Flash и Flex и только потом пытаться строить AIR-приложения. Изложение исходит из того, что у читателя есть подготовка в области Flash и/или Flex.

Структура книги

Книга устроена необычайно просто. Это сравнительно небольшое издание, состоящее всего из восьми глав. Поэтому мы не сочли необходимым

деление на тематические части. В главе 1 читатель знакомится с тем, что представляет собой AIR, а также получает необходимые в дальнейшем базовые сведения. Глава 8 связывает все воедино, рассказывая, как реально строить, развертывать и обновлять AIR-приложения. В каждой из остальных шести глав описывается какая-то одна из логических групп AIR API. Например, в главе 2 рассказывается о доступе к локальной файловой системе, включая чтение и запись файлов.

Представление кода

Книга изобилует примерами кода – от коротких фрагментов до полных приложений. Весь код представлен моноширинным шрифтом, что выделяет его среди остального текста. Кроме того, многие крупные фрагменты кода приводятся в виде нумерованных листингов с заголовками. На эти листинги всегда ссылается близлежащий текст, и они часто снабжены аннотациями.

Загрузка кода

Практически все приведенные в книге листинги можно загрузить с сайта книги на *www.manning.com/AdobeAIRinAction*.

Форум Author Online

Приобретая «Adobe AIR in Action», вы получаете бесплатный доступ к закрытому форуму, организованному Manning Publications, на котором можно комментировать книгу, задавать технические вопросы и получать помощь от авторов и других пользователей. Регистрация в форуме и доступ к нему осуществляются с *www.manning.com/AdobeAIRinAction*. На этой странице рассказано, как попасть на форум после регистрации, какого рода помощь доступна и как вести себя на форуме.

Manning предоставляет читателям место, где может происходить содержательное общение читателей между собой и с авторами книги. При этом не оговорен какой-либо минимальный объем участия со стороны авторов, чей вклад в работу форума является добровольным (и бесплатным). Поэтому мы советуем задавать авторам каверзные вопросы, чтобы их интерес не иссяк!

Форум Author Online и архивы предыдущих дискуссий сохраняются на веб-сайте издательства, пока распространяется книга.

О названии

Объединяя первое знакомство, обзоры и примеры конкретных задач, книги серии «In Action» способствуют обучению и запоминанию. Согласно данным науки о познании лучше всего запоминается то, что открыто во время самостоятельного исследования.

В Manning нет специалистов по когнитивистике, но мы считаем, что знание закрепляется тогда, когда оно проходит через стадии изучения, применения и, что любопытно, пересказа изученного. Понять и запомнить новые вещи, т. е. овладеть ими, можно только после их активного изучения. Люди учатся на практике. Важным аспектом руководств «In Action» является их ориентированность на примеры. Они поощряют читателя к самостоятельным экспериментам с новым кодом и исследованию новых идей.

Есть и другая, более приземленная причина выбора именно такого названия книги: наши читатели – занятые люди. Книги нужны им для того, чтобы выполнить задание или решить проблему. Им нужны такие книги, которые предоставляют удобный доступ к нужной информации и позволяют узнать то, что нужно, и тогда, когда нужно. Для таких читателей и предназначены книги данной серии.

Об обложке

Фигура на обложке «Adobe AIR in Action» – «Провинциальный юрист, депутат от провинций». Иллюстрация взята из путеводителя начала 19 века «L'Encyclopedie des Voyages», напечатанного во Франции. В то время путешествовали ради удовольствия лишь немногие, и такие путеводители были популярны; они знакомили как путешественников, так и тех, кто не покидал своего кресла, с жителями других районов Франции, с ее воинами, чиновниками, аристократами, а также с людьми, населяющими заморские страны.

Разнообразие рисунков, помещенных в «Encyclopedie des Voyages», ярко свидетельствует об уникальности и своеобразии городов и местностей, существовавшем какие-то 200 лет назад. В те времена «дресс-код» двух местностей, располагавшихся на расстоянии всего нескольких десятков километров друг от друга, однозначно определял, откуда человек родом. Этот путеводитель напоминает о существовавшем в ту эпоху ощущении изолированности и удаленности – равно как в любые иные исторические эпохи, исключая нашу нынешнюю, сверхподвижную.

С той поры дресс-коды изменились, и различия по регионам, столь отчетливые в прежние времена, ушли в прошлое. Сейчас бывает даже трудно отличить обитателей одного континента от другого. Если встать на более оптимистическую точку зрения, то мы, возможно, променяли культурное и зрительное разнообразие на более диверсифицированную личную жизнь. Или более диверсифицированную и интересную интеллектуальную и техническую жизнь.

Мы в Manning полагаем, что книжная обложка, отражающая разнообразие местной жизни двухсотлетней давности, возвращенное к жизни картинками из этого путеводителя, подходит для выражения нашего восхищения изощренностью, оригинальностью и прихотливостью компьютерных технологий.

В этой главе:
- Элементы Adobe AIR
- Дескрипторы приложений AIR
- Создание новых проектов AIR
- Компиляция приложений AIR

1

Введение в Adobe AIR

Работа с HTML, Flash, Flex и тысячами других технологий имеет одно общее свойство: все получающиеся приложения используют технологии, предназначенные для Web. Это замечательно, если ваша цель – создать веб-приложение, но никуда не годится, если вам нужно настольное приложение. Интегрированная среда выполнения Adobe (Adobe integrated runtime – AIR) решает эту проблему. Благодаря Adobe AIR вы можете применить свое умение создавать веб-приложения во Flash и Flex (а также HTML и JavaScript) для создания настольных приложений. Это создает заманчивую перспективу.

Каждый полет начинается с подготовки к взлету. Путешествие по Adobe AIR пройдет аналогичным образом. Для начала мы совершим общий обзор AIR, а потом детально изучим применение Flex и Flash для создания AIR-приложений. Мы рассмотрим следующие необходимые базовые понятия, лежащие в основе работы с AIR:

- Различные части Adobe AIR, включая исполнительную систему, средства инсталляции и AIR-приложения, а также существующие между ними связи.
- Проблему безопасности и аутентификации приложений, в том числе цифровые подписи. Вы узнаете, что такое цифровая подпись, какие типы подписей существуют, и чем следует руководствоваться при их выборе.
- Основные этапы создания AIR-приложений с помощью Flex Builder, Flash CS3 или Flex 3 SDK.

Покончив с предисловиями, перейдем к выяснению, в чем, собственно, сущность AIR.

1.1. Анатомия Adobe AIR

Adobe AIR позволяет разработчикам веб-приложений применить имеющиеся у них навыки для создания настольных приложений. Опираясь на свое знание HTML, JavaScript, Flash и Flex, вы можете создавать приложения, которые могут выполняться на настольных системах с помощью исполнительной среды (runtime environment) без необходимости компиляции для конкретных целевых операционных систем. В этом разделе мы определим, что такое исполнительная среда и обсудим, в каких случаях может понадобиться создать настольное приложение. Затем мы расскажем, как могут пригодиться для этого те навыки, которые у вас уже есть.

1.1.1. Разработка приложений для исполнительной среды

Если вы работаете на компьютере под Windows, то, несомненно, вам приходится запускать множество .exe-файлов. Файл .exe представляет собой скомпилированное приложение, которое может подавать команды непосредственно той системе, на которой оно выполняется. Это означает, что .exe-файл (или эквивалентный ему) обладает преимуществом относительной самодостаточности. Однако при этом возникает определенное препятствие, связанное с тем, что необходимо скомпилировать приложение в определенном формате, специфичном для данной платформы. Это означает, что при таком подходе вы должны создать версии вашего приложения «только для Windows» или «только для OS X». Этапы традиционного подхода к созданию приложений следующие:

1. Написать код на выбранном языке.
2. Скомпилировать код в формат, который может непосредственно выполняться в конкретной операционной системе.
3. Запустить скомпилированное приложение.

Более гибкий способ – использовать исполнительную среду вместо того, чтобы компилировать программу для каждой операционной системы. Такой метод исполнительной среды применяется на многих распространенных платформах приложений, включая Java и .NET; он же принят в Adobe AIR. При использовании исполнительной среды процесс создания приложения будет таким:

1. Написать код на выбранном языке.
2. Скомпилировать код в промежуточный формат.
3. Запустить код промежуточного формата в исполнительной среде.

Исполнительные среды дают разработчикам возможность, написав код один раз, выполнять его на любом компьютере, работающем под управлением любой операционной системы, если только в ней установлена требуемая исполнительная среда. Исполнительная среда представляет собой библиотеку, непосредственно выполняемую в данной операцион-

ной системе. Исполнительная среда выступает в качестве посредника для выполняемых в ней программ. Благодаря обеспечению такого уровня абстракции в отношении запускаемых в исполнительной среде программ и операционной системы появляется теоретическая возможность создания для разных компьютеров исполнительных систем, которые будут выполнять одно и то же приложение одинаково на различных платформах.

Какое отношение имеет это все к Adobe AIR? Как уже отмечалось, AIR является исполнительной средой. При создании AIR-приложения вы компилируете его и упаковываете в промежуточный формат, называемый .air-файлом. .air-файл и его содержимое нельзя установить или запустить на компьютере, на котором не была предварительно установлена исполнительная среда AIR. Если среда AIR установлена, файл .air позволяет запустить приложение как на машине с Windows, так и на машине под OS X. Это большое благо для разработчика приложения.

Следует сказать, что у веб-приложений есть преимущества перед обычными настольными приложениями. В каких же случаях тогда вообще может возникнуть необходимость создавать настольные приложения? Раз вы читаете эту книгу, то, вероятно, у вас есть некие основания, а мы приведем свои, представляющиеся нам важными, мотивации.

1.1.2. Зачем нужны настольные приложения?

Веб-интерфейс электронной почты позволяет вам читать свою корреспонденцию с любого компьютера, подключенного к Интернету. Это пример одного из главных достоинств веб-приложений, заключающегося в том, что они не привязаны к конкретной машине. Кроме того, веб-приложения:

- позволяют легко развертывать обновления и новые версии вашего программного обеспечения;
- обычно обеспечивают определенный уровень защиты для пользователей, поскольку на них оказывают влияние средства защиты броузера и плеера (например, Flash Player);
- позволяют распределять вычисления, выполняя их частично на машине клиента, а частично – на сервере.

Однако у веб-приложений есть недостатки. Два самых существенных состоят в том, что они:

- не имеют такого же доступа к функциям операционной системы, как настольные приложения;
- требуют, чтобы компьютер был подключен к Интернету; это недостаток, если нужно работать с приложением, когда вы находитесь в самолете или где-то на природе.

Приложения AIR сочетают в себе лучшие качества как веб-приложений, так и настольных приложений. Поскольку приложения AIR основаны на технологиях веб-приложений, вам (как разработчику)

крайне просто получить доступ к сетевым ресурсам и частично или полностью интегрировать существующие веб-приложения. Но поскольку AIR-приложения выполняются локально, у них есть доступ к системным ресурсам, которого обычно нет у веб-приложений. Это означает доступность таких возможностей, как перетаскивание файлов между AIR-приложениями и файловой системой, обращение к локальным базам данных и – что, может быть, важнее всего – создание эффективных условий работы пользователя при непостоянном подключении к сети – как в режиме онлайн, так и автономно. Приложения AIR также предоставляют функции контроля за появлением обновлений, благодаря которым пользователи всегда могут быть уверены, что работают с самой свежей версией программы (эта тема обсуждается в главе 8).

Другой вопрос, на который хотелось бы получить ответ, – зачем могут понадобиться веб-технологии при создании настольных приложений. Самая очевидная причина – имеющееся у разработчика знание веб-технологий, которое хотелось бы применять разными способами. Если вы сможете создавать настольные приложения с помощью уже имеющихся навыков, вам не придется изучать новый язык и новые технологии, только чтобы сделать настольное приложение. Но есть и другие причины, по которым вы можете захотеть создавать настольные приложения с помощью веб-технологий. Веб-технологии лучше всего приспособлены для создания приложений, которые используют сетевые ресурсы. В условиях, когда настольным приложениям все чаще требуется поддержка интерактивности и доступ к Интернету, предпочтительнее создавать такие приложения с помощью языков, специально разработанных для работы в сети. Другое преимущество применения HTML, JavaScript, Flash и Flex для разработки настольных приложений состоит в том, что эти языки обычно значительно превосходят традиционные языки в возможностях создания запоминающихся, привлекательных и интересных интерфейсов пользователя.

1.1.3. Изучаем возможности AIR

AIR предоставляет разработчикам веб-приложений множество замечательных возможностей для создания настольных приложений. Но что именно вас ждет? Сейчас мы представим основные возможности, предоставляемые AIR. Подробности вы узнаете дальше, по мере чтения книги.

Все, что позволено при создании веб-приложений, позволено также при создании AIR-приложений. Это происходит благодаря включению в AIR движка WebKit (того же, что применен в броузере Safari) и Flash Player. Поэтому можно пользоваться теми же базовыми функциями ActionScript и JavaScript, что и при развертывании веб-приложений. Кроме того, становится доступен специфический API для AIR. В него входят функции, приведенные в табл. 1.1.

Таблица 1.1. Категории функций API, специфичные для AIR

Функция	Описание
Интеграция с файловой системой	AIR разрешает системные операции с файловой системой, включая чтение, запись и удаление.
Перетаскивание (drag-and-drop)	Пользователи могут перетаскивать файлы и каталоги из операционной системы в приложение AIR.
Копирование и вставка	Пользователи могут применять функции копирования-вставки операционной системы для копирования данных между приложениями AIR и операционной системой.
Локальные базы данных	AIR-приложения могут создавать локальные базы данных и подключаться к ним.
Аудио	Приложения AIR на основе HTML легко могут использовать звук.
Встроенный HTML	Приложения AIR на основе Flex и Flash могут показывать в своих объектах HTML и JavaScript.

В AIR-приложениях поддерживается доступ ко всем этим функциям. Однако для их использования AIR-приложения должны выполняться в исполнительной среде, которая их поддерживает. В следующем разделе мы посмотрим, как выполнять приложения AIR.

1.2. Выполнение AIR–приложений

Для создания приложения AIR используется набор инструментов AIR, в качестве которого может выступать Flex Builder 3, AIR SDK и т. п., после чего файлы приложения упаковываются в файле .air. Подробнее об упаковке в файле .air вы узнаете в разделе 1.4 этой главы. Пока вам достаточно знать, что файл .air – это единственный файл, который нужно передать тому, кто захочет установить ваше приложение. Наряду с термином «.air-файл» используется термин «установочный файл» (installer file).

Имея на руках .air-файл, вы можете передать его всем, у кого на компьютере уже есть исполнительная среда AIR, и они смогут легко установить вашу программу. Когда у пользователя установлена среда AIR, ему достаточно сделать двойной щелчок по .air-файлу, который он получил от вас или загрузил из Интернета.

Если на компьютере пользователя нет AIR, ему придется сначала установить эту среду. Для установки AIR есть два пути:

- *Ручная установка* – осуществляется путем загрузки программы установки для конкретной ОС (Windows или OS X) и ее запуска.
- *Автоматическая установка* – требует от разработчика публикации в сети файла .swf (называемого «значком» – badge), при щелчке по которому происходит установка вашего приложения. Если

у пользователя уже стоит AIR, у него сразу установится ваше приложение. Если нет – он сможет сначала установить AIR.

Примечание

Подробнее о распространении приложений AIR, в том числе об автоматической установке, можно узнать в главе 8.

Каким бы образом пользователь ни начал устанавливать приложение AIR – двойным щелчком по присланному ему файлу .air или щелчком по значку установки на веб-странице, – перед ним предстанет ряд стандартных экранов помощника установки. На рис. 1.1 приведен пример того, как может выглядеть первый этап. После установки приложения AIR на машине пользователя его можно запустить, как всякое другое приложение, – выполнив двойной щелчок по значку приложения на рабочем столе или выбрав его в меню.

Выяснив, как запускать AIR-приложения, можно перейти к вопросу их создания.

Рис. 1.1. *При установке приложения AIR отображается экран установки AIR с информацией о приложении и его авторе*

1.3. Безопасность и аутентичность приложений AIR

Мы были бы не правы, если бы при знакомстве с Adobe AIR упустили две связанных между собой проблемы: безопасности и аутентичности. Это важные для разработчика приложения вопросы, поскольку связанные с ними упущения или ошибки могут иметь пагубные последст-

вия. Поэтому необходимо знать, какие средства защиты и проверки подлинности приложений предоставляет AIR, и каковы должны быть ваши действия для защиты пользователей ваших приложений.

1.3.1. Безопасность приложений AIR

Один из флагманских продуктов Adobe – Flash Player, успех которого, в частности, определяется чрезвычайными мерами, предпринятыми Adobe (а до того Macromedia), чтобы гарантировать невозможность умышленного или неумышленного нанесения разработчиками Flash-приложений вреда компьютеру пользователя. У Flash Player есть множество функций безопасности, нацеленных на защиту пользователей. Благодаря им пользователь не должен испытывать беспокойство при просмотре Flash-содержимого в сети

Приложения AIR являются настольными, а потому они должны иметь больше возможностей доступа к компьютерной системе пользователя, чем сетевые Flash-приложения. Хотя приложения AIR также выполняют Flash-контент, у *этого* Flash-контента больше возможностей нанести урон системе пользователя, чем у контента Flash, находящегося в сети. Это определенный компромисс: расширение набора функций влечет за собой повышение риска.

Приложения AIR выполняются с помощью посредника – исполнительной среды. Поэтому Adobe может в значительной мере управлять тем, что позволено или не позволено делать приложению AIR. Однако, несмотря на то, что среда исполнения в значительной мере уменьшает риски, AIR предоставляет своим приложениям значительно больше привилегий, чем могло быть доступно соответствующим веб-аналогам.

Первое, что должен осознать разработчик AIR-приложений, – это необходимость проявить уважение к пользователям своего продукта, серьезно относясь к проблемам безопасности. Например, важно тщательно следить за всеми параметрами, получаемыми кодом, который выполняется в вашем приложении. Не разрешайте пользователям вводить произвольные значения и не допускайте, чтобы динамические, полученные из сети значения участвовали в качестве параметров для кода, который осуществляет такие операции, как, скажем, обращение к файловой системе. Существует подробный документ от Adobe по проблемам безопасности, который находится по адресу: download.macromedia.com/pub/labs/air/air_security.pdf.

1.3.2. Гарантии аутентичности приложения

Чтобы избавить от тревог пользователей вашего приложения, Adobe требует наличия цифровой подписи у каждого приложения AIR. (Заметьте, что подпись требуется только при создании инсталлятора, а разработку и тестирование своих приложений вы можете проводить без всяких подписей.). Цифровая подпись дает пользователю возможность проверить два свойства приложения: подлинность и целостность.

Цифровая подпись моделирует обычную подпись чернилами на листе бумаги, подтверждая подлинность издателя приложения (аутентичность), и отсутствие в приложении изменений, сделанных после публикации (целостность). Слежение среды AIR за целостностью можно проиллюстрировать простым примером. Можете сами убедиться в том, что исполнительная среда AIR откажется устанавливать модифицированный .air-файл. Для этого вам понадобятся файл .air и утилита zip. Файл .air имеет формат архива, который может прочесть любая утилита zip. Сделайте следующее:

1. Запустите файл .air и убедитесь, что сначала среда AIR предложит вам установить приложение. Не нужно щелкать по кнопке Install, когда она появится в помощнике установки. Достаточно проверить, что среда AIR предлагает вам возможность установки.
2. Щелкните по кнопке Cancel и выйдите из помощника установки.
3. С помощью утилиты zip добавьте в архив какой-нибудь файл. Годится любой файл: можете создать пустой текстовый файл и добавить его в архив. Если вы работаете в Windows, проще всего поменять расширение имени файла с .air на .zip, перетащить текстовый файл в архив .zip и затем вернуть прежнее расширение .air.
4. Запустите файл .air. На этот раз вы получите сообщение об ошибке, говорящее о том, что файл .air поврежден и не может быть установлен.

Приложения AIR поставляются с цифровыми подписями и с цифровыми сертификатами. Есть два основных вида сертификатов: подписанные самостоятельно и выпущенные сертифицирующими органами. У тех и других есть свои достоинства и недостатки.

Преимущество подписанных самостоятельно сертификатов в том, что их просто изготовить. Обновление Flash CS3 AIR и Flex 3 SDK (а затем и Flex Builder 3) предлагают средства для изготовления сертификатов ваших приложений AIR, заверенных личной подписью. Подробнее о том, как изготавливать такие сертификаты, будет рассказано далее в этой главе.

Самостоятельно подписанные сертификаты предоставляют пользователям определенный уровень безопасности, поскольку проверяют целостность приложения. Однако они практически бесполезны для проверки личности того, кто опубликовал приложение. Это примерно то же, что самому нотариально заверять подлинность своих документов. Поэтому, когда сертификат подписан самим изготовителем, помощник установки указывает, что его личные данные не известны. Очевидно, это недостаток, поскольку пользователь не может чувствовать себя в безопасности, и он будет менее склонен устанавливать приложение, когда производитель неизвестен, чем когда личные данные производителя можно проверить. Сертифицирующий орган – это организация, выпускающая сертификаты и действующая в качестве независимой стороны при проверке ваших личных данных.

Сертифицирующий орган выдает сертификаты только после проверки ваших личных данных, обычно на основании таких документов, как выпущенное государством удостоверение личности. Преимущество такого официально заверенного сертификата в том, что он дает больше уверенности в ваших подлинных данных, чем подписанный самостоятельно. Когда сертификат выпущен сертифицирующим органом, помощник установки Adobe показывает данные этого документа как данные издателя программы. С другой стороны, очевидны некоторые неудобства: получать сертификат у уполномоченного органа дольше и труднее, чем подписывать самостоятельно. Кроме того, сертифицирующие органы обычно берут плату за свои услуги. (На момент написания книги самый дорогой сертификат для подписи кода AIR-приложений стоил 299 долл. США.)

Два наиболее известных издателя сертификатов – это VeriSign (*www.verisign.com*) и thawte (*www.thawte.com*), хотя формально владельцем thawte теперь является VeriSign. Если вы хотите обеспечить высший уровень сертификации для своего приложения AIR, вам нужно приобрести сертификат в одном из этих центров. Вам потребуется так называемый сертификат для подписи кода (codesigning certificate). Подробнее об условиях приобретения сертификатов можно узнать на вебсайтах центров сертификации.

Примечание

Помимо VeriSign и thawte, существуют другие сертифицирующие организации, – в том числе некоммерческие, такие как CAcert.org, – которые могут предоставить сертификат для подписи программы. Перед приобретением того или иного сертификата вам следует провести собственные изыскания (CAcert.org заставит вас немало побегать, прежде чем выдать вам сертификат разработчика) и убедиться, что выбранный сертификат будет принят на большинстве компьютеров. Если сертификат не будет принят, то приложение AIR по-прежнему будет оцениваться как выпущенное неизвестным автором. Если вы не до конца определились с выбором, поговорите с кем-нибудь в организации, выдающей сертификаты, и задайте интересующие вас вопросы.

Приступая к созданию AIR-приложений, вы, возможно, не захотите тратить средства на покупку сертификата только для того, чтобы собрать несколько примеров и разослать инсталляторы своим друзьям. Не забывайте, что сертификат нужен только тогда, когда вы хотите создать инсталлятор. Тестировать свои приложения AIR можно и без сертификата. Однако, когда вы будете готовы создать для своего приложения файл .air, нужно продумать, какую цифровую подпись выбрать для приложения. Связать сертификат с приложением можно только один раз. Это означает, что нельзя сначала воспользоваться сертификатом с собственной подписью, а потом заменить его на сертификат, выданный сертифицирующим центром. Если вы сделаете новую подпись с другим сертификатом, то пользователи более старых версий вашего приложения не смогут его обновить.

1.4. Создание приложений AIR

Теперь, когда вы знаете о том, что такое AIR, из чего состоит AIR, как запускать приложения AIR и как обеспечиваются безопасность и аутентичность приложений AIR, почти все готово для создания приложения AIR. Фактически именно об этом будет рассказано в нескольких последующих разделах этой главы. Вы даже сможете построить несколько простых приложений AIR, чтобы прочувствовать, что вас ожидает в дальнейшем, после чтения книги. Прежде чем бросаться в неизведанное, мы потратим некоторое время на изучение предстоящего маршрута, чтобы дать вам представление о том, что вас ожидает.

Способов создания приложений AIR много. В табл. 1.2 приведено краткое описание набора инструментов.

Таблица 1.2. Инструменты Adobe AIR

Название	Тип исходного кода приложения AIR	Бесплатный	Описание
Flex Builder 3	Flex/ActionScript	Нет	Коммерческое средство для создания веб- и AIR-приложений на базе Flex. Сам инструмент построен на базе Eclipse. Flex Builder 3, автоматизирует и упрощает создание приложений AIR.
Flex 3 SDK	Flex/ActionScript	Да	Бесплатный SDK, включающий в себя все компиляторы и инструменты Flex Builder 3, но без средств автоматизации и графического интерфейса пользователя Flex Builder.
Flash CS3 с обновлением AIR	Flash/ActionScript	Нет	Flash CS3 поставляется без средств AIR. Однако бесплатное обновление для Flash CS3 позволяет строить приложения AIR прямо из среды разработки Flash.
Dreamweaver CS3 с расширением AIR	HTML/JavaScript	Нет	Коммерческий редактор HTML с расширением AIR, которое в значительной мере автоматизирует создание приложений AIR.
AIR SDK	HTML/JavaScript	Да	Бесплатный SDK со всеми инструментами командной строки, нужными для создания приложений AIR на базе HTML/JavaScript.

Для полноты картины мы включили в табл. 1.2 также инструменты HTML/JavaScript для создания приложений AIR. Однако, повторимся, в этой книге мы фокусируем внимание исключительно на применении Flex и Flash в качестве средств создания приложений AIR. В разделах 1.6, 1.7 и 1.8 вы сможете подробнее узнать, как создавать приложения AIR с помощью Flex Builder, Flash и Flex SDK, соответственно.

Примечание

Все инструменты AIR можно загрузить с сайта Adobe (*www.adobe.com/go/air*).

В разделе 1.5 вы узнаете о дескрипторах приложений. Понимание дескрипторов – важный элемент для получения общей картины создания приложений AIR. Несмотря на то, что многие инструменты AIR (Flex Builder и Flash CS3 с обновлением AIR) автоматически создают файл дескриптора для проекта, все же полезно ознакомиться с тем, как выглядит дескриптор и что в нем содержится. Рекомендуем целиком прочесть раздел 1.5, прежде чем переходить к 1.6, 1.7 или 1.8. Однако если вам не терпится поскорее начать строить приложение и вы проскочите вперед, мы никому об этом не скажем.

Сейчас мы перейдем к дескрипторам приложений. После прочтения следующего раздела переходите к разделу, в котором обсуждается тот набор инструментов, с помощью которого вы будете строить свои приложения AIR.

1.5. Знакомство с дескрипторами приложений AIR

Независимо от того, с помощью каких инструментов вы строите приложение AIR, вам необходимо создать дескриптор приложения. Некоторые инструменты автоматически генерируют базовый дескриптор, но необходимо разбираться в том, что собой представляет этот дескриптор и как им пользоваться. Дескрипторы приложений AIR являются файлами XML, в которых описываются приложения AIR. При создании из приложения AIR пакета для распространения в дескриптор помещаются некоторые сведения, необходимые инструментам AIR для правильной сборки приложения. В состав этих сведений входят, в частности, уникальный идентификатор приложения, номер версии и те данные, которые показываются в процессе инсталляции. Ниже приводится пример того, как может выглядеть простой файл дескриптора. Обратите внимание, что все файлы дескрипторов начинаются с объявления XML (`<?xml version="1.0" encoding="utf-8" ?>`).

```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/application/1.0.M4">

   <id>com.manning.books.airinaction.Example</id>

   <version>1.0</version>
```

```
    <filename>ExampleApplication</filename>

    <initialWindow>
       <content>ExampleMain.swf</content>
    </initialWindow>

</application>
```

Если вы хотите сразу начать строить приложение AIR, можете приступать. Приведенного примера вам будет достаточно для создания элементарного приложения AIR. Если сейчас вы проскочите вперед, вы сможете позже вернуться к этому разделу и более подробно ознакомиться с дескрипторами.

В следующих ниже разделах детально описываются элементы, составляющие файл дескриптора.

1.5.1. Элемент application

Элемент `application` является обязательным, и это корневой элемент файла дескриптора. Элемент `application` требует атрибута `xmlns`. Атрибут `xmlns` определяет пространство имен дескриптора. Значение пространства имен всегда заранее определено, и для каждого приложения, которое вы создаете для определенной версии AIR, это значение всегда одно и то же. Для AIR 1.0 значением пространства имен должно быть `http://ns.adobe.com/air/application/1.0.M4`. Пространство имен указывает, какая версия AIR необходима для выполнения приложения. Каждая новая версия AIR использует новое пространство имен.

Дополнительно можно задать атрибут `minimumPatchLevel`. С его помощью можно потребовать, чтобы пользователь установил определенное обновление среды AIR, прежде чем запускать ваше приложение. Этот атрибут не обязателен. Пользоваться им следует только тогда, когда вам известно, что для корректной работы вашего приложения необходимо определенное обновление.

Поскольку элемент `application` является корневым в файле дескриптора, все последующие элементы оказываются дочерними для него. Единственными обязательными элементами являются следующие четыре: `id`, `version`, `filename` и `initialWindow`.

1.5.2. Элемент id

Элемент `id` служит уникальным идентификатором приложения. В каждый данный момент в системе может быть установлено только одно приложение с данным идентификатором. Идентификатор приложения составлен из идентификатора издателя (который берется из сертификата, применяемого при публикации файла .air) и значения элемента `id`. Это означает, что, строго говоря, значение элемента `id` должно быть уникальным только среди всех приложений данного издателя. Хотя это не является абсолютно необходимым, но мы считаем удобным применять глобально уникальные `id`, использующие извест-

ный принцип обратных доменных имен. В приводившемся примере простого дескриптора таким идентификатором является `com.manning.books.air.Example`. Глобальную уникальность здесь обеспечивает `com.manning`, что является обратным от `manning.com`. Длина `id` может составлять от 1 до 212 символов, которыми могут быть только буквы, цифры, точки и дефисы.

1.5.3. Элемент version

Элемент `version` позволяет задать номер версии вашего приложения. AIR никак не интерпретирует номер версии приложения, но с его помощью вы можете программным образом проверить, что у пользователя стоит новейшая версия вашего приложения. Поскольку AIR не пытается каким-либо образом обрабатывать номер версии, можно воспользоваться любым строковым значением. Обычно версии обозначаются числами, например, 1.0 или 2.5.1, но могут включать в себя и буквы, обозначающие номер ревизии, например, 4.0a.

1.5.4. Элемент filename

В элементе `filename` указывается имя .air-файла. Значение имени файла используется также в качестве имени приложения (во время инсталляции), если не задан элемент `name`.

Элемент `filename` должен содержать только символы, допустимые в именах файлов, и не должен включать в себя расширение. Значение `filename` не должно также оканчиваться точкой.

1.5.5. Элемент initialWindow

Элемент `initialWindow` сообщает о том, какой фактический контент (файл .swf или .html) нужно использовать для сборки приложения. Элемент `initialWindow` служит контейнером для дополнительных элементов. Единственным обязательным дочерним элементом является content, указывающий, какой файл .swf (или .html) нужно использовать. Ниже показан простейший элемент `initialWindow`:

```
<initialWindow>
   <content>ExampleMain.swf</content>
</initialWindow>
```

Дополнительно элемент `initialWindow` может содержать следующие необязательные элементы:

- `systemChrome` – это значение указывает, должно ли окно, в котором показывается приложение, использовать оформление (рамку и панель заголовка), установленные в операционной системе. Если задать значение `standard`, будет использовано стандартное системное оформление. Если задать `none`, стандартное оформление системы использовано не будет. Для приложений AIR на базе Flex компоненты Flex имеют специальный вид, когда атрибут `systemChrome` имеет значение `none`.

- transparent – это булево значение указывает, должно ли происходить альфа-смешение окна приложения с рабочим столом (т. е. можно ли видеть рабочий стол через полупрозрачное окно). Если задать значение true, то можно пользоваться альфа-эффектами, но помните, что при этом потребуется больше ресурсов и отображение приложения может замедлиться. Кроме того, если вы хотите задать для transparent значение true, то нужно задать для systemChrome значение none.
- visible – это булево значение указывает, должно ли окно приложения быть видимым с самого начала. Обычно для этого атрибута указывается значение false, когда вы не хотите показывать окно, пока программным образом не зададите в коде самого приложения место и размеры окна. Затем вы можете программно управлять видимостью окна.
- height – высота окна приложения в пикселах.
- width – ширина окна приложения в пикселах.
- minimizable, maximizable, resizable – эти элементы позволяют указать, можно ли во время выполнения приложения минимизировать, максимизировать и изменять размеры его окна. По умолчанию имеют значения true.
- x, y – координаты x и y начального положения окна.
- minSize, maxSize – минимальный и максимальный допустимый размер окна при попытке изменить его размер.

Вот пример элемента initialWindow, для которого установлено большинство этих значений:

```
<initialWindow>
   <content>ExampleMain.swf</content>
   <systemChrome>none</systemChrome>
   <transparent>true</transparent>
   <height>500</height>
   <width>500</width>
   <minimizable>false</minimizable>
   <maximizable>false</maximizable>
   <resizable>false</resizable>
   <x>0</x>
   <y>0</y>
</initialWindow>
```

Как уже было сказано, единственным обязательным элементом initialWindow является content. Все другие можно опустить, и тогда будут использованы значения по умолчанию.

1.5.6. Элемент name

Элемент name находится на одном уровне с initialWindow, т. е. является дочерним для тега application. Значение name определяет каталог, в ко-

тором приложение устанавливается по умолчанию. Также, значение `name` отображается в заголовке окна приложения во время его работы. Кроме того, `name` появляется на первом экране инсталлятора, как видно на рис. 1.1. Если значение `name` не указано, вместо него используется значение `filename`.

1.5.7. Элементы title и description

Элементы `title` и `description` – одного ранга с `initialWindow`, т. е. должны быть дочерними для тега `application`. Оба они не обязательны и задают строки, показываемые инсталлятором. Элемент `title` определяет, что показывается в заголовках инсталлятора, как на рис. 1.1 и 1.2. Строка `description` выводится на втором экране инсталлятора, как на рис. 1.2.

Элементы `title` и `description` участвуют только в установке и никогда не появляются в работающем приложении.

1.5.8. Элемент installFolder

Элемент `installFolder` является необязательным и определяет название подкаталога по умолчанию, в который устанавливается приложение. Пользователь всегда может изменить этот каталог во время установки приложения AIR. Однако с помощью `installFolder` можно задать подкаталог, который будет частью значения по умолчанию, как на рис. 1.2.

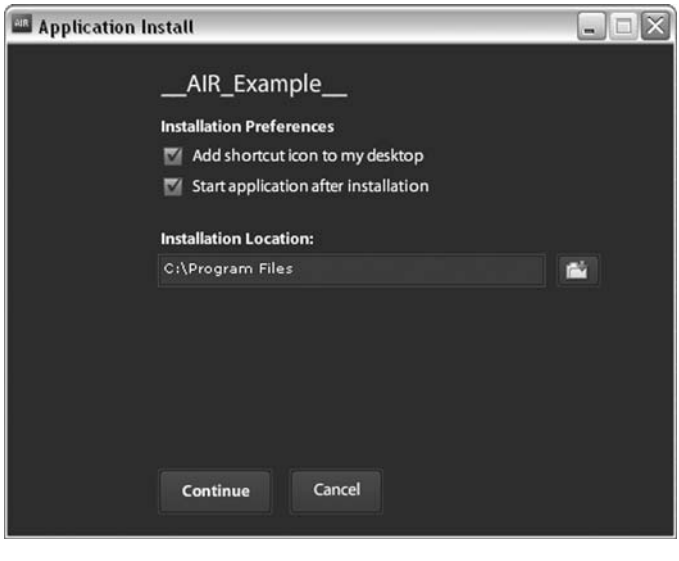

Рис. 1.2. *Второй экран инсталлятора приложения AIR, на котором пользователь может задать параметры установки*

Примечание

Нельзя изменить имя главного каталога в каталоге по умолчанию, используемого приложениями AIR.

В Windows этим каталогом всегда будет Program Files на главном диске; в OS X это всегда /Applications. Однако элемент `installFolder` позволяет изменить используемый подкаталог. Например, если задать для `installFolder` значение `ExampleInc/ExampleApplication`, то на машине под Windows приложение будет установлено в `Program Files\ExampleInc\ExampleApplication`, а на машине под OS X приложение будет установлено в `/Applications/ExampleInc/ExampleApplication`.

Элемент `installFolder` должен быть дочерним для тега `application`. Это необязательный элемент. Если опустить его, то приложение будет установлено в подкаталог, определяемый значением элемента `filename`.

1.5.9. Элемент programMenuFolder

Элемент `programMenuFolder` используется только в Windows и игнорируется другими операционными системами. Этот элемент позволяет задать имя папки, из которой берется ярлык программы для пункта меню All Programs в меню Start.

1.5.10. Элемент icon

По умолчанию приложения AIR используют стандартные значки AIR для размещения на рабочем столе, в меню Start, панели задач и т. д. Однако можно задать свои значки, использовав для этого элемент `icon` в XML-файле дескриптора. У элемента `icon` должно быть четыре дочерних элемента с именами `image16x16`, `image32x32`, `image48x48` и `image128x128`. В каждом из этих дочерних элементов прописывается путь к графическому файлу. Сами графические файлы должны иметь формат .png и быть включены в приложение AIR. Вот пример элемента `icon`:

```
<icon>
   <image16x16>icon16.png</image16x16>
   <image32x32>icon32.png</image32x32>
   <image48x48>icon48.png</image48x48>
   <image128x128>icon128.png</image128x128>
</icon>
```

Если ваши значки имеют непрямоугольную форму, не забудьте сохранить прозрачность при записи файлов .png.

1.5.11. Элемент customUpdateUI

Если в дескрипторе есть элемент `customUpdateUI`, то приложение получает возможность самостоятельно обрабатывать действия по своему обновлению. (Подробнее эта процедура описана в главе 8.) Если вы хо-

тите, чтобы приложение само управляло процессом обновления, присвойте этому элементу значение `true`. Если его значение `false` или опущено, для обновления используются стандартные диалоги AIR.

1.5.12. Элемент fileTypes

Необязательный элемент `fileTypes` позволяет связать с приложением некоторые типы файлов. Если некий тип файла связан с приложением, то при двойном щелчке по значку файла этого типа автоматически запускается приложение AIR, если оно не было запущено ранее. После этого генерируется событие, которое перехватывается работающим приложением AIR и сообщает ему сведения о файле, по которому был сделан двойной щелчок, после чего приложение AIR решает, как ему действовать дальше. Подробнее об обработке этого события можно прочесть в главе 3.

Если есть элемент `fileTypes`, у него должен быть один или несколько дочерних элементов `fileType`. Каждый элемент `fileType` должен содержать элементы `name` и `extension`. Кроме того, элемент `fileType` может содержать элементы `description` и `contentType`. Значением `contentType` может быть тип MIME. (Подробнее о типах MIME можно прочесть на en.wikipedia.org/wiki/MIME.) Ниже следует пример элемента `fileTypes`, в котором зарегистрирован единственный тип файла:

```
<fileTypes>
   <fileType>
      <name>com.manning.ExampleApplicationSavedSettings</name>
      <extension>exp</extension>
      <description>A saved settings file for Example Application
    ➥</description>
      <contentType>text/xml</contentType>
   </fileType>
</fileTypes>
```

Как видите, в этом примере для обеспечения уникальности имени использовано значение `name` в виде обратного доменного имени. Обратите также внимание на отсутствие точки перед расширением имени.

1.6. Создание приложений AIR с помощью Flex Builder

Flex Builder 3 предоставляет встроенные функции разработки приложений AIR и содержит все необходимые инструменты AIR. Если вы хотите строить AIR-приложения, основанные на Flex, то Flex Builder 3 предоставляет для этого отличные возможности.

В нескольких следующих разделах вы познакомитесь с основами применения Flex Builder 3 для создания AIR-приложений. В частности, вы узнаете, как сконфигурировать новый проект AIR, создать для

проекта MXML и другие файлы, выполнить тестирование/отладку проекта и создать инсталлятор приложения.

1.6.1. Конфигурирование нового проекта AIR

Если вы хотите начать разработку нового приложения AIR с помощью Flex Builder 3, то прежде всего необходимо создать проект AIR. Чтобы создать проект AIR, следует выбрать в меню Flex Builder 3 пункты File→New→Flex Project. В результате откроется диалоговое окно New Flex Project, приведенное на рис. 1.3.

Помощник – тот же самый, с помощью которого в Flex Builder 3 создается новый Flex-проект, поэтому для уточнения необходимых деталей вы можете обратиться к справочнику по Flex Builder. Единственное отличие – выбор «Desktop application» в качестве типа приложения, как показано на рис. 1.3.

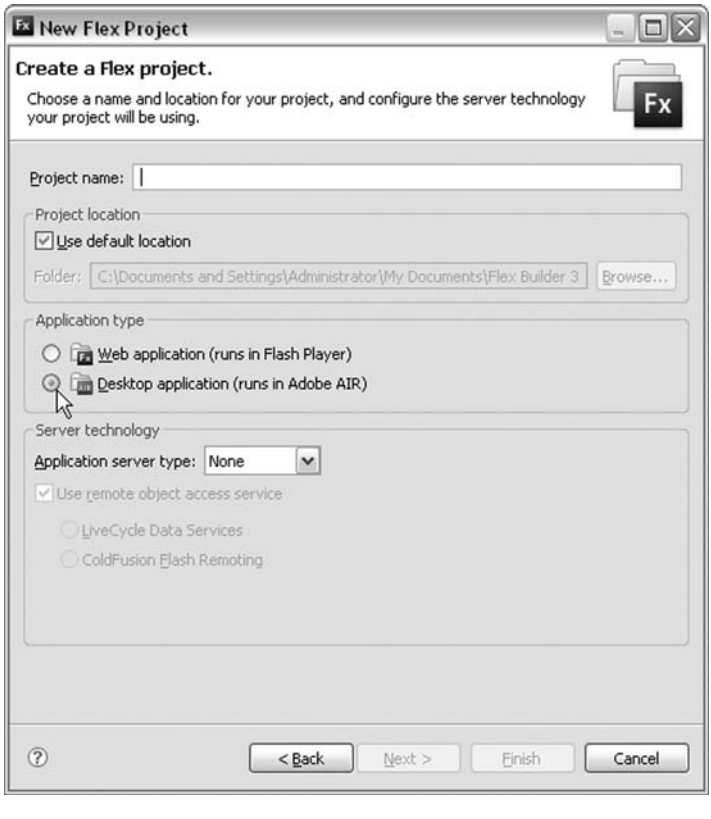

Рис. 1.3. *В диалоговом окне создания нового проекта Adobe AIR в Flex Builder 3 нужно прежде всего указать имя проекта и тип приложения. Для приложений AIR укажите в качестве типа «desktop application» (настольное приложение)*