Содержание

Введение

С выходом Windows Presentation Foundation (новая система разработки графических интерфейсов пользователей для персональных приложений Windows) в 2006 г. и Silverlight в 2008 г. процесс разработки клиентских приложений изменился в лучшую сторону. Корпорация Майкрософт приняла смелое решение распрощаться с некоторыми концепциями и технологиями, которые оставались неизменными с самой первой версии Windows, и заменить их чем-то более новым и совершенным. Конечно, идти в ногу со всеми новинками, предлагаемыми разработчикам ПО, нелегко, но на этот раз стоит постараться. Майкрософт поставила на Silverlight и WPF очень много, тут просто не может быть неудачи. Именно в этих технологиях – будущее разработки клиентских приложений.

Silverlight реализована на многих платформах в виде надстройки над браузером. И такие надстройки скоро окажутся на большинстве клиентских компьютеров, обращающихся к Интернету, поскольку процедура их развертывания не сложнее загрузки обычного веб-контента, а раздавать их способен любой веб-сервер без какой бы то ни было дополнительной инфраструктуры. Благодаря богатству графического интерфейса и простоте обращения к удаленным службам Silverlight станет одним из основных игроков на поле *обогащенных интерактивных приложений* (rich interactive applications – RIA). Кроме того, Silverlight открывает путь к технологии разработки клиентских приложений Windows Presentation Foundation, за которой Майкрософт видит будущее программирования для персональных компьютеров.

В Сети веб, где в настоящее время главную роль играет Adobe Flash, Silverlight представляет не просто альтернативу. В ее основе лежит .NET! Любой программист, знакомый с .NET, воспримет Silverlight как дом родной, поскольку и библиотеки, и языки программирования (C#, VB.NET, Ruby, Python), и среда разработки (Visual Studio, Expression Studio) те же самые. К тому же программистам для Silverlight доступны все новые концепции, возникшие и отточенные при создании Windows Presentation Foundation: привязка к данным, отделение поведения от представления, элементы управления, не имеющие собственного внешнего вида, но открытые для стилизации и применения шаблонов с помощью таких могучих инструментов, как Expression Blend, развитая система анимации, интеграция с мульмедийным содержимым и т. д. Новый язык разметки XAML (XML-based Application Markup Language), разработанный Майкрософт, может выступать в качестве моста между программистами и дизайнерами, что открывает совершенно новые возможности работы над проектами.

Эта книга не является и не задумывалась как исчерпывающее руководство по платформе Silverlight. Честно говоря, я даже не уверен, что подобная книга нужна – в вашем распоряжении весь Интернет, где можно найти более полную и актуальную справочную информацию, чем в любой книге. Я лишь хотел помочь вам увидеть, почему программирование – это удовольствие, а Silverlight – даже больше чем удовольствие. Я стремился заразить вас вирусом Silverlight. Сложные идеи объясняются просто, а многочисленные примеры и рисунки помогут как начинающим, так и искушенным разработчикам.

О примерах кода

Мы стремились придерживаться единого стиля форматирования, чтобы код выглядел так же, как в Visual Studio.

Строки нумеруются только в случае необходимости, например когда в тексте имеются ссылки на конкретные строки.

Все исходные тексты можно загрузить из Сети по адресу www.galasoft.ch/SL2U/Code. Благодаря стараниям технического редактора книги Дж. Бойда Нолана примеры переведены также на язык VB.NET.

Старше на год

Я начал работать над этой книгой в сентябре 2007 г., а сейчас я стал ровно на год старше. С профессиональной точки зрения, это был самый интересный год в моей жизни. С 1996 г. мне довелось работать с самыми разными клиентскими технологиями и языками, в том числе C, VB, Java, HTML, CSS, JavaScript, ASP.NET, Windows Forms и, наконец, Windows Presentation Foundation и Silverlight. Но никогда новая платформа программирования так не радовала меня. Писать книгу трудно, это большая работа. Но работа над этой книгой оказалась такой приятной и интересной, что я ни разу не пожалел о своем решении. Если бы была возможность все повторить, я бы подписал контракт без колебаний. А теперь, когда книга вышла из печати, я с нетерпением жду, какие применения Silverlight найдут читатели. Разработка программного обеспечения сродни искусству, а Silverlight – самая богатая палитра, какую только можно вообразить. Ну так берите книгу, садитесь за компьютер и начинайте проектировать и писать код. Покажите миру, на что вы способны. Я буду ждать.

Приятного кодирования!

Лоран

Глава 1. Введение в Silverlight

Все началось с того, что в 2006 г. на конференции MIX в Лас-Вегасе корпорация Майкрософт представила восторженной толпе графических дизайнеров, программистов и бизнесменов новый революционный каркас для разработки пользовательских интерфейсов – *Windows Presentation Foundation*. Была также организована дополнительная демонстрация, на которой речь шла о менее известной технологии с варварски звучащим названием – *Windows Presentation Foundation/Everywhere*, или *WPF/E*. Смотреть еще было особо не на что, но анонс завораживал: «WPF/E позволит создавать обогащенные, интерактивные приложения, которые смогут работать во всех основных браузерах на большинстве платформ, не исключая и мобильные устройства».

Спустя год с небольшим на второй конференции MIX Скотт Гатри (Scott Guthrie) (генеральный директор Майкрософт, в подчинении которого находится большинство коллективов, работающих над .NET) поднялся на сцену и предложил аудитории потрясающую демонстрацию. Кошмарной аббревиатуры WPF/E не стало, ее место заняло название Silverlight (рис. 1.1).

На экранах крутился новый яркий логотип. Повсюду были разбросаны градиенты и анимации. В окне браузера летали самолеты, соединяя американские города, которые Скотт планировал посетить. Встроенный в браузер движок JavaScript играл в шахматы против .NET, наглядно демонстрируя превосходство откомпилированной .NET-программы над интерпретируемым JavaScript-кодом. Вы не ослышались, браузер исполнял .NET! Причем не только Internet Explorer, но и Firefox! И даже на компьютере Macintosh.

Тот еще выдался часок! В тот же день, но чуть позже, была продемонстрирована новая программа Expression Encoder, позволяющая создавать видеоролики и прокручивать их в браузере, предварительно скопировав на потоковый (или непотоковый) веб-сервер. Более того, с помощью Expression Encoder вы можете полностью изменить внешний облик видеоплеера, взяв за основу готовые шаблоны и модифицировав все, что угодно, – от цветов до геометрических форм – в Expression Blend.

Рис. 1.1. Логотип Microsoft Silverlight

Примечание

Записи всех демонстраций, представленных на конференциях MIX 2006, 2007 и 2008, можно посмотреть по адресу http://sessions.visitmix.com.

Эти первые три демонстрации стали лишь прелюдией к трем удивительным дням, целиком и полностью посвященным Silverlight. Меньше чем за год Silverlight превратилась из мало кому известного второстепенного аттракциона в могучую технологию с огромным потенциалом.

Истоки технологии Silverlight

Возможно, вам будет интересно узнать о происхождении технологии Silverlight и о том, каково ее место в ряду многих других имеющихся технологий создания пользовательских интерфейсов. Не исключено, что излагаемые ниже факты вам хорошо известны, – это зависит от вашего прошлого опыта. В таком случае можете спокойно пропустить последующие разделы (первым интересным для вас разделом, наверное, станет «Работа на разных платформах»). Если же вам хочется освежить память, то прочитайте этот и несколько последующих разделов, в которых мы дадим краткий обзор истории со времен зарождения Всемирной паутины до чудес сегодняшнего дня.

Веб-страницы существуют уже давно. Язык HTML был представлен миру еще в 1992 г. (Вы знаете, что один год жизни собаки принято считать равным семи годам человеческой жизни? Так вот, если считать в годах компьютера, то 1992 для человека все равно, что Средневековье!) Разумеется, поначалу возможности HTML были очень ограничены. Простенькие веб-страницы содержали преимущественно текст и гиперссылки (в этом и состояла революционность HTML, из-за чего он получил название *язык разметки гипертекста*).

Но очень скоро изобретатели HTML осознали, что содержимое веб-страниц должно быть более разнообразным. И первым шагом в этом направлении стало появление тега IMG, позволявшего размещать на странице изображения. Тем самым основы обогащенного содержимого были заложены, и первый популярный браузер Mosaic уже умел отображать такие страницы.

Автоматизация веб-страниц с помощью языка JavaScript

Следующим большим шагом в погоне за обогащенным контентом стало включение в браузер движка JavaScript.

Этот язык программирования (впервые в 1995 г. он появился в браузере Netscape, а в 1996 г. – и в Internet Explorer) позволял организовывать гораздо более интересные взаимодействия с пользователем. Раньше пользователь мог лишь отправить *запрос* веб-серверу, который этот запрос обрабатывал и возвращал результаты в виде *ответа*. Понятно, что такая модель не обеспечивала высокой скорости, особенно во времена коммутируемых соединений.

Примечание

У этого языка сценариев есть несколько названий. Компания Netscape называла его JavaScript, Майкрософт – JScript, а в стандарте он именуется ECMAScript. В этой книге мы будем пользоваться названием JavaScript.

Предупреждение

Контроль на стороне клиента легко обойти, поэтому *всегда* необходимо проверять данные также и на сервере!

С появлением JavaScript стало возможно обрабатывать данные прямо на стороне клиента, не посылая запросов серверу. Первым и самым очевидным применением клиентских технологий стала предварительная проверка данных, за счет чего удавалось избежать отправки бесполезной информации, то есть напрасного потребления ресурсов Сети и сервера. С помощью сценария на JavaScript можно было обнаружить ошибки на ранней стадии, поэтому некорректные данные вообще не покидали клиентский компьютер. Разумеется, если избежать взаимодействия с сервером было невозможно, например для получения данных, выполнения сложных вычислений, аутентификации пользователей и т. д., запрос все же приходилось посылать. Такая модель распространена и на многих современных сайтах.

Каскадные таблицы стилей

Еще одним значительным продвижением в сторону обогащенного веб-контента явилось изобретение *каскадных таблиц стилей* (CSS). Наконец-то стало возможно отделить содержимое от представления. Информация о внешнем облике страницы изымалась из самой страницы и оформлялась в виде правил, хранящихся во внешних файлах. У такой модели был целый ряд достоинств:

- разделение обязанностей между различными коллективами в зависимости от квалификации и специализации. Графические дизайнеры отвечают за внешний вид, программисты – за функциональность, а отдел маркетинга – за содержимое;
- в небольших проектах разработчик может сначала сосредоточить усилия на структуре и функциональности страницы, а позже – когда ее содержимое станет понятно – вернуться к оформлению. Это гораздо эффективнее, чем делать все одновременно;
- внешний вид страницы можно изменять, не трогая саму страницу;
- одни и те же стили можно использовать многократно, не переписывая код. CSS-классы можно поместить в какое-то центральное хранилище и *ссылаться* на них из любого места. На самом деле можно даже хранить стили на выделенном сервере и обращаться к ним с других веб-серверов.

По мере усложнения CSS-стилей страницы становились все более и более изысканными. Доходило даже до того, что отношение сторон страницы оказывалось не менее важным, чем содержимое. Можно бесконечно спорить о том, хорошо ли это (в конце концов, пользователь все-таки приходит за контентом, разве не так?), но если при прочих равных условиях контент

Предупреждение

Хотя CSS позволяет создавать красивые страницы, с его помощью можно сформировать и уродливых монстров. Увы, технология лишена вкуса, им наделен (или не наделен) лишь программист или дизайнер. То же самое верно и в отношении Silverlight (да и любой другой технологии пользовательских интерфейсов).

можно оформить красиво или скучно, то, конечно, с симпатичной страницей работать будет приятнее.

Переход к DHTML

По мере совершенствования технологий JavaScript и CSS появилась возможность взаимодействия между ними. Так родилась на свет аббревиатура *DHTML* (*Dynamic HTML*); это комбинация JavaScript и CSS, используемая в контексте HTML-страниц.

С помощью языка JavaScript стало возможно обращаться к элементам, размещенным на странице (содержимому и стилям), и модифицировать их. Поскольку это клиентская технология, никаких обращений к серверу при этом не требовалось.

Так как CSS позволяет задавать степень прозрачности элемента, то можно создавать эффекты постепенного появления и исчезновения изображения. Возможность устанавливать абсолютную позицию элемента означает, что элемент можно перемещать по странице. Помимо «простого» добавления логики и привлекательности, такое взаимодействие между JavaScript и CSS позволяет создавать обогащенные страницы, с которыми удобнее работать.

Хотя звучит все это заманчиво, у DHTML есть и ограничения:

- прозрачность поддерживается не всеми браузерами, а те, что ее поддерживают, применяют различный синтаксис, что превращает сопровождение кода в кошмар;
- скорость перемещения элементов по экрану относительно невелика, поэтому анимация получается не такой плавной, как хотелось бы;
- невозможно поворачивать элементы. Если вы хотите создать вращающуюся кнопку, придется имитировать эффект с помощью нескольких изображений, и плавности не получится;
- повернуть текст (например, в заголовке, логотипе и т. д.) можно, лишь представив его в виде картинки. Но в таком случае сильно осложняется задача локализации, поскольку картинки приходится создавать для каждого поддерживаемого языка;
- «активная область» любого элемента прямоугольна. Иными словами, даже если вы нарисовали круглую кнопку с прозрачным фоном, активировать (нажать) ее можно щелчком за пределами круга. Курсор примет форму руки, хотя указывает на прозрачную часть занимаемой кнопкой области.

К счастью, технология Silverlight устраняет все эти недостатки, а поскольку созданные с ее помощью элементы можно смешивать со стандартными элементами HTML, то открывается реальная возможность графически обогатить веб-страницу.

AJAX — более развитый способ коммуникации

Язык JavaScript не только лег в основу технологии DHTML, но и принес важнейшее усовершенствование в плане коммуникации между клиентом и сервером, которое положительно отразилось на удобстве работы пользователей. Речь идет о технологии *Asynchronous JavaScript and XML* (*AJAX*). Не вдаваясь в детали, ска-

Совет

Писать программы с использованием AJAX нелегко. Но позже вы увидите, что и в этой области Silverlight 2 предлагает немало улучшений, существенно упрощая фоновые коммуникации.

жем, что AJAX позволяет посылать из JavaScript-сценария запросы веб-серверу и получать от него ответы, не перезагружая страницу целиком. Коммуникация происходит в фоновом режиме, и пользователь ее даже не замечает.

Эта технология позволяет получать от сервера дополнительную информацию, не покидая текущего контекста. Поэтому пользователь не «травмируется» перезагрузками страницы и не видит в промежутке «белого экрана».

Использование сторонних надстроек

Принимая во внимание ограничения DHTML, разработчики придумали разнообразные надстройки над браузером, цель которых – сделать работу пользователя комфортнее. К числу наиболее известных относятся Java-апплеты, элементы управления ActiveX и Flash-приложения. В этом разделе мы дадим беглый обзор достоинств и недостатков этих технологий, чтобы понять, как они соотносятся с Silverlight.

Примечание

У всех рассматриваемых в этом разделе технологий имеется общий досадный изъян – невозможность поместить HTML-контент поверх области, занятой подключаемым модулем. И Java-апплет, и элемент ActiveX, и Flash-приложение всегда располагаются на переднем плане. Silverlight решает эту проблему, позволяя смешивать контент обоих типов.

Java-апплеты

Java-апплеты были какое-то – недолгое – время очень популярны, но потом их популярность сошла на нет. Еще можно встретить страницы, на которых применяется Java, но их становится все меньше. Основная проблема – очень большое время инициализации Java. Когда загружается страница, содержащая Java-апплет, долгое время ожидания становится раздражающим фактором.

С другой стороны, Java – замечательный язык программирования. Он проложил дорогу .NET, и многие из самых удачных черт C# имеют корни в Java. Ко всему прочему, на нем можно программировать интересные графические эффекты, например знаменитый эффект ряби на воде.

Между Java-апплетом и JavaScript-сценарием возможно взаимодействие, хотя и ограниченное. Однако такой интерфейс поддерживают не все браузеры, в которых реализована поддержка Java. Кроме того, синтаксис довольно сложный, а порядок взаимодействия запутанный.

Java открывает возможность для альтернативных способов коммуникации с веб-сервером. Например, сервер может общаться напрямую с клиентом, что при классическом способе коммуникации в веб невозможно. Хотя в некоторых случа-

ях эта технология может оказаться весьма полезной, при определенных обстоятельствах она представляет угрозу безопасности. Из-за этого крупные корпорации, серьезно относящиеся к безопасности, неохотно внедряли у себя Java-апплеты, что в конце концов и «убило» Java в браузерах.

Примечание

В любой книге по веб-технологиям должно быть четко сказано: Java и JavaScript не имеют между собой ничего общего. Программируя на JavaScript, вы *не* пишете программу на Java.

Элементы управления ActiveX

Когда в 1993 г. корпорация Майкрософт представила миру технологию COM, стало возможным создание так называемых элементов ActiveX, которые инкапсулировали некую логику и взаимодействовали с внешним миром посредством COM-интерфейсов. Это позволило разрабатывать элементы управления, применяя классические технологии Windows (в частности, на языке C++), и загружать их в браузер. Разрешен даже ограниченный доступ из элемента к JavaScript, а с его помощью – и к веб-странице.

Основной недостаток элементов ActiveX состоит в том, что в них используется устаревшая технология 15-летней давности. К тому же поддерживает их только браузер Internet Explorer на платформе Windows.

Flash-приложения

Adobe Flash, пожалуй, является сегодня самой популярной сторонней надстройкой. К достоинствам следует отнести возможность создания развитых графических эффектов, существуют даже сайты, целиком написанные на Flash. Хотя Flash-контент часто называют «Flash-роликом», правильнее было бы говорить о «Flash-приложениях».

Основной недостаток Flash – сложность программирования. Для создания пользовательского интерфейса необходим специальный (коммерческий) редактор. Кроме того, «застраничный» код можно писать только на языке ActionScript, являющемся подмножеством JavaScript. Flash создавался прежде всего с расчетом на графических дизайнеров, поэтому программистам довольно трудно с помощью прилагаемого инструментария создавать выразительный контент и функциональность.

Предупреждение

Программирование сайта целиком на Flash (или на Silverlight) не назовешь удачной идеей. На многих платформах (особенно на мобильных телефонах, КПК и т. д.) доступ к таким сайтам невозможен вовсе или сильно ограничен. Подобные технологии призваны обогатить содержимое сайта, а не полностью заменить его.

Примечание

Майкрософт не рекламирует Silverlight как «убийцу» Flash. Такая стратегия в любом случае была бы обречена на провал, если учесть, сколько дизайнеров работают с Flash и сколько уже написано Flash-контента. Поскольку обе технологии могут мирно уживаться друг с другом, то и нет необходимости развязывать религиозные войны по поводу того, какая лучше!

Посредством JavaScript можно организовать ограниченное взаимодействие между Flash-приложением и объемлющей его веб-страницей.

Работа на разных платформах

При разработке новых веб-технологий необходимо стремиться к охвату как можно большего количества платформ. Веб, по определению, существует всюду; можно встретить устройства с доступом в Интернет самых разных форм и возможностей. Хорошим примером может служить технология Adobe Flash: надстройки для нее реализованы для многих браузеров в самых разных операционных системах. Это и делает Flash такой мощной платформой. Корпорация Майкрософт прекрасно понимает, что и Silverlight должна работать в разных браузерах.

Как уже было сказано, на момент написания этой книги среда исполнения Silverlight была реализована для Internet Explorer и Firefox в ОС Windows и Firefox и Safari в ОС Macintosh. Готовится версия для Linux. Поддержать одинаковые интерфейсы на всех платформах, где работает Silverlight, – очень непростая задача. До сих пор с ней удавалось справляться, и это крупное достижение в борьбе за единообразное веб-окружение.

Примечание

Чтобы Silverlight одинаково работала на многих платформах, Майкрософт сотрудничает с компанией Novell и разработчиками каркаса .NET с открытыми исходными текстами под названием «Mono».

Уже представлены прототипы Silverlight для смартфонов, но пока мало что известно о том, какие функции будут поддерживаться на этих устройствах с ограниченными возможностями. В конце 2008 г. должна выйти первая версия Silverlight для мобильных телефонов (на платформе Windows Mobile и телефонах Nokia). Она будет поддерживать спецификацию Silverlight 1, включая и видео.

Поскольку Silverlight должна работать на столь разнородных платформах, неизбежны какие-то функциональные ограничения (по сравнению с полной платформой .NET). Следует принимать во внимание многие факторы:

- ставится цель максимально уменьшить размер среды исполнения, чтобы ее можно было без труда загрузить через Интернет. В обозримом будущем размер не должен превышать 5 Мб. Если сравнить это с размером полной среды исполнения .NET (даже убрав из нее все относящееся к серверному программному обеспечению), становится понятно, что реализовать удастся далеко не все;
- функции, требующие аппаратной акселерации (особенно трехмерную анимацию), было бы слишком сложно реализовать одинаково на разных платформах (не говоря уже о мобильных устройствах);

Совет

Говоря об аппаратной акселерации, имеют в виду, что сложные вычисления выполняются специализированными графическими процессорами (а не программно). Такие процессоры работают гораздо быстрее, чем любая программная реализация.

- на разных платформах применяются различные графические технологии, и не каждая позволяет реализовать все требуемые эффекты. И так-то удивительно, насколько одинаково работают версии Silverlight на платформах Windows и Macintosh. Чтобы у Silverlight было будущее, абсолютно необходимо сохранять полную совместимость на всех поддерживаемых платформах и браузерах.

Вышеупомянутые причины породили ожесточенные споры о том, что включать в Silverlight, а что – нет. Будет очень интересно наблюдать за дальнейшим развитием этой платформы!

Совет

Майкрософт прислушивается к вам! Не стесняйтесь обращаться к евангелистам и другим разработчикам Silverlight как на специализированных форумах (http://silverlight.net/forums), так и через их блоги (см. раздел «Чтение блогов, посвященных Silverlight» в главе 24). Расскажите им, как вы применяете технологию и чего от нее ждете. Сторонние разработчики оказали влияние на эволюцию WPF, то же справедливо и для Silverlight! Пусть они услышат ваш голос!

Веб-приложения и безопасность

Добиться, чтобы веб-приложение было безопасным, нелегко. Хотя Майкрософт чаще, чем любая другая компания, становилась объектом ожесточенной критики за недостаточную безопасность приложений и операционных систем, надо быть честными: любая популярная веб-технология небезупречна, везде были проблемы с безопасностью (Java, Firefox, операционная система Linux, новый браузер Google Chrome и многие другие программы становились мишенями для атак).

Майкрософт подходит к проблеме безопасности приложений на основе Silverlight очень серьезно. Такие приложения работают в «песочнице», которая ограничивает доступные им функции и защищает компьютер от атак. Всякий раз, как принимается решение открыть в песочнице новую функцию, тщательно проверяется, не создаст ли это изменение брешь для потенциальной атаки.

Если бы человечество было по природе своей добродетельно, то, конечно, программировать было бы проще. Но безопасность связана не только с атаками злоумышленников. К этой же теме относятся плохо написанный код, порча памяти и т. д. На наше счастье, .NET предоставляет безопасную платформу для программирования (так называемый «управляемый код»). В отличие от неуправляемого кода на C++, многие ошибки изничтожены в зародыше, благодаря самой природе управляемых языков программирования. Порча памяти почти невозможна. А поскольку неиспользуемая память автоматически освобождается *сборщиком мусора*, то и утечки гораздо менее вероятны.

Конечно, было бы наивно полагать, что Silverlight никогда не будут атаковать или что написанное для Silverlight приложение не может завершиться аварийно. Но благодаря опыту, накопленному командами, работающими над .NET, благода-

ря управляемым языкам программирования и тому особому вниманию, которое Майкрософт уделяет этой проблеме, можно надеяться, что Silverlight станет очень безопасным окружением.

Знакомство с Silverlight.net

На сайте сообщества http://silverlight.net имеется немало полезной информации о Silverlight, включая вводные пособия и учебные руководства, примеры и т. д. Пожалуй, наибольший интерес представляет галерея, расположенная по адресу http://silverlight.net/community/communitygallery.aspx.

В ней вы найдете множество примеров, созданных Майкрософт и сторонними разработчиками. Это неплохая отправная точка для тех, кто хочет получить представление о том, на что способна Silverlight. В настоящее время галерея разбита на два раздела: Silverlight 1.0 (на основе JavaScript) и Silverlight 2.0 (на основе .NET). Но даже если вы установите Silverlight 2 (мы рекомендуем поступить именно так), то все равно сможете исполнять более старые приложения.

Что необходимо для запуска Silverlight?

Silverlight – это надстройка над браузером. Она устанавливается отдельно и расширяет функциональность веб-страниц. В настоящее время она реализована для браузеров Internet Explorer и Firefox в операционных системах Windows XP и Vista. Существуют также версии для Firefox и Safari на платформе Macintosh. Во время написания этой книги велась работа над версией для Linux – в соответствии с историческим соглашением между Майкрософт и Novell.

Для запуска приложений Silverlight вам потребуется совместимый веб-браузер. Если с помощью браузера, поддерживающего Silverlight, зайти на страницу, где имеется приложение Silverlight, то вместо результата работы приложения вы увидите значок Install Microsoft Silverlight, показанный на рис. 1.2.

Рис. 1.2. Значок Install Microsoft Silverlight

- Щелкните по этому значку, чтобы перейти на страницу сайта Майкрософт, откуда можно загрузить и установить среду исполнения Silverlight.
- После установки Silverlight в Internet Explorer даже не потребуется перезагружать браузер. В Firefox такая необходимость может возникнуть, но это небольшая проблема.

Альтернативно можно установить Silverlight 2 со страницы http://silverlight.net/GetStarted.

Предупреждение

Если вы работали с Silverlight раньше и на вашем компьютере уже установлена версия, более старая, чем необходима данному приложению, то вы также увидите значок «Install Microsoft Silverlight».