Programming ASP.NET AJAX

Christian Wenz

O'REILLY®

Программирование в ASP.NET AJAX

Кристиан Венц

Санкт-Петербург — Москва
2008

Кристиан Венц

Программирование в ASP.NET AJAX

Перевод А. Киселева

Главный редактор	*А. Галунов*
Зав. редакцией	*Н. Макарова*
Научный редактор	*Б. Попов*
Редактор	*Ю. Бочина*
Корректор	*С. Минин*
Верстка	*Д. Орлова*

Венц К.

Программирование в ASP.NET AJAX. – Пер. с англ. – СПб: Символ-Плюс, 2008. – 512 с., ил.

ISBN-13: 978-5-93286-122-6
ISBN-10: 5-93286-122-3

Книга Кристиана Венца «Программирование в ASP.NET AJAX» – полное практическое введение в новую платформу Microsoft ASP.NET AJAX 1.0, которая предоставляет большие преимущества при разработке Ajax-приложений, схожие с теми, которые дает ASP.NET для разработки серверных сценариев. Вы узнаете, как с помощью технологий Ajax без особых затрат времени создавать профессионально выполненные динамические веб-страницы.

После общего обзора платформы и основ JavaScript и Ajax рассмотрена организация составляющих частей .NET, включая пакеты Extensions, Control Toolkit, Futures CTP и Microsoft AJAX Library. Также читателю предлагается исчерпывающая глава об элементе управления UpdatePanel, который позволяет организовать независимое обновление отдельных частей веб-страницы.

Насыщенная примерами, демонстрирующими ключевые аспекты платформы, эта книга идеально подходит не только разработчикам ASP.NET, стремящимся расширить свои возможности за счет использования Ajax, но и всем специалистам, кто интересуется данной платформой, независимо от того, какие технологии они используют в настоящее время.

ISBN–13: 978–5–93286–122–6
ISBN–10: 5–93286–122–3
ISBN 0–596–51424–7 (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 31.07.2008. Формат 70×100^{1}/$_{16}$. Печать офсетная.
Объем 32 печ. л. Тираж 2000 экз. Заказ №
Отпечатано с готовых диапозитивов в ГУП «Типография «Наука»
199034, Санкт-Петербург, 9 линия, 12.

Оглавление

Предисловие

В Википедии (на странице *http://en.wikipedia.org/wiki/Ajax*) дается более 40 толкований слова Ajax – имена персонажей из поэмы Гомера «Илиада» (Аякс Великий и Аякс Малый), название футбольной команды из Амстердама, пара моделей автомобилей, кличка лошади и даже мое любимое чистящее средство, выпускаемое компанией Colgate. Но кроме всего прочего, под термином Ajax подразумевается целый набор технологий, которые, по мнению многих, стали революционными во Всемирной паутине. Если верить различным публикациям в Интернете, Ajax – это будущее веб-разработки, основа Web 2.0 и действенное средство против множества проблем.

Многие веб-разработчики стремятся обеспечить своих клиентов приложениями, обладающими богатыми возможностями, не создавая при этом обычные приложения Windows (или, по определенным причинам, не имея возможности их создавать). Технология Ajax предоставляет все необходимое для этих целей. Она позволяет наделять их возможностями, которые раньше были характерны только для обычных настольных приложений, такими как горячие комбинации клавиш и техника «перетащить-и-отпустить» (drag-and-drop).

Платформа ASP.NET под кодовым названием «Atlas», разрабатываемая компанией Microsoft, обеспечивает набор средств, реализующих функциональность Ajax, для разработчиков ASP.NET. Она предлагает большую часть функциональности, доступной для разработки ASP.NET-приложений на стороне сервера. Осенью 2006 года было объявлено о выходе окончательной версии продукта под названием ASP.NET AJAX. (Впрочем, название Atlas более удобно для произношения.)

Я долго воздерживался от написания книги об Ajax. В течение многих лет я использовал на практике и писал о технологиях, являющихся составными частями Ajax, но сам термин появился лишь в начале 2005 года, еще до того, как данная технология стала популярной. На мой взгляд, лучше всего по этому поводу выразился Клеменс Вастерс (Clemens Vasters) в своем блоге, озаглавив эту запись «Web 2.0 бла-бла-бла AJAX бла-бла-бла преимущества!(?)» (*http://vasters.com/clemensv/PermaLink,guid,d88c1112-d8da-496e-9fd0-8cf03cf55c32.aspx*).

Крикливая реклама, которая сейчас развернулась вокруг Ajax, напоминает мне шумиху вокруг XML и веб-служб, поднятую несколько лет тому назад: все говорили о них, но мало кто читал соответствующие

спецификации. После того как шумиха улеглась и появились реальные приложения, оказалось, что объединение этих технологий дало существенный эффект.

Я уверен, что развитие Ajax пойдет тем же путем, только более высокими темпами. Во Всемирной паутине уже сейчас можно обнаружить удобные Ajax-приложения.

Но вернемся к моему нежеланию писать об Ajax.

Я всегда говорил, что описание самой технологии Ajax займет от силы страниц 20–30. Добавив немного вводной информации и примеров, этот объем можно довести до 75 страниц, может быть, даже до 100. Но чем можно было бы заполнить остальную часть книги? Авторам многих книг, посвященных Ajax, которые сейчас можно найти в продаже, пришлось пойти на определенные ухищрения, чтобы достигнуть желаемого количества страниц.

Мое мнение изменилось, когда в сентябре 2005 года я посетил конференцию профессиональных разработчиков Microsoft (Microsoft Professional Developers Conference) в Лос-Анджелесе и впервые увидел Atlas. На этой конференции компания Microsoft объявила о платформе, которая не просто обеспечивает функциональные возможности Ajax, но и предоставляет инструментальные средства, облегчающие разработку современных веб-приложений. Тогда я подумал, что это действительно стоит того, чтобы написать книгу. Я приступил к работе над рукописью, основываясь на предварительной версии Atlas. Мне пришлось несколько раз переписывать ее по мере появления промежуточных версий платформы Atlas, которые попадали мне в руки. Нехватка документации к промежуточным версиям вынуждала меня выполнять самостоятельные исследования внутренних механизмов Atlas. Как результат этих усилий, в этой книге можно найти несколько неофициальных способов решения некоторых задач.

В сентябре 2006 года вышла книга «Programming Atlas». Как одна из первых книг, посвященных этой теме, она содержала подробные сведения о платформе, которая еще продолжала изменяться. В конце января 2007 года вышла окончательная версия ASP.NET AJAX 1.0. Кроме изменения названия, существенные изменения претерпела и внутренняя организация платформы, что потребовало создания новой редакции книги, так как все существующие приложения необходимо было адаптировать под новую версию платформы.

В данной книге рассказывается о том, как создавать профессиональные динамические веб-страницы на основе платформы ASP.NET AJAX. В самом начале книги приводятся некоторые вводные сведения о JavaScript и ASP.NET, поскольку они будут необходимы для понимания темы.

Я стараюсь придерживаться принципа «больше дела, меньше слов». Поэтому книга содержит большое количество примеров, иллюстрирующих ключевые аспекты использования платформы ASP.NET AJAX.

Кроме того, стараясь фокусировать внимание на конкретных фактах, я стремился создавать небольшие примеры, объясняющие один или два вопроса, благодаря чему мне удалось избежать в книге длинных листингов, каждый из которых объяснял бы сразу множество вопросов. На своем опыте автора книг и преподавателя я уже убедился, что более короткие примеры приводят к лучшим результатам и существенно упрощают изучение материала.

Кроме того, следует отметить, что все примеры являются достаточно универсальными. Это позволит вам добавлять их непосредственно в свои собственные проекты, видоизменять и извлекать из них отдельные фрагменты по мере потребностей. Все примеры являются совершенно самостоятельными, что облегчает их использование.

Для кого написана книга

Данная книга адресована двум группам веб-разработчиков: тем, кто уже использует ASP.NET и хотел бы привнести в свои приложения возможности, которые предоставляет технология Ajax, и тем, кто в настоящее время использует в своей работе другие технологии, но заинтересован в освоении платформы ASP.NET AJAX. Кроме того, книга будет интересна программистам на JavaScript, которые хотели бы избежать головной боли при разработке программного кода, совместимого с разными типами броузеров. В этой книге используются языки программирования C# и JavaScript. Если вам необходимы начальные сведения об этих языках, можно порекомендовать обратиться к книгам «Learning C# 2005» (авторы: Джесс Либерти (Jesse Liberty) и Брайан Макдональд (Brian MacDonald)) и «Learning JavaScript» (автор Шелли Пауэрс (Shelley Powers)).

Как устроена книга

Глава 1 «ASP.NET AJAX, Ajax и ASP.NET» содержит обзор технологии Ajax и платформы ASP.NET AJAX, а также охватывает процедуру установки ASP.NET AJAX, описывает ее структуру и содержит первый пример.

Глава 2 «JavaScript» представляет собой краткое введение в JavaScript. Несмотря на то, что платформа ASP.NET AJAX стремится избавить программиста ASP.NET от ненужных функциональных подробностей, тем не менее некоторое знание JavaScript будет действительно необходимо настоящему специалисту в ASP.NET AJAX.

Глава 3 «Ajax» описывает технологии без лишней рекламы. Здесь вы узнаете о том, как работают внутренние механизмы Ajax; и все, что действительно необходимо знать, описывается менее чем на 20 страницах.

Глава 4 «Использование расширений JavaScript в ASP.NET AJAX» рассказывает о том, как ASP.NET AJAX обогащает функциональность

клиентского JavaScript за счет добавления объектно-ориентированных возможностей и переопределения некоторых классов платформы .NET для обеспечения возможности их использования на стороне клиента.

Глава 5 «Веб-службы» рассказывает о веб-службах, использующих XML. Хотя в платформе ASP.NET AJAX основной упор сделан на разработке клиентских приложений, в ней также имеются средства разработки веб-служб, исполняемых на стороне сервера. Сюда включаются возможности обработки ошибок и поддержка сеансов.

Глава 6 «UpdatePanel: обновление только части страницы» представит вашему вниманию элемент управления `UpdatePanel`, который позволяет обновлять отдельные части страницы без необходимости выполнять обновление всей страницы целиком. Это один из наиболее важных элементов платформы ASP.NET AJAX.

Глава 7 «Использование службы управления профилями в ASP.NET AJAX» описывает порядок организации доступа из JavaScript к прикладному интерфейсу ASP.NET 2.0 Profile API в ASP.NET AJAX.

Глава 8 «Использование службы аутентификации в ASP.NET AJAX» описывает функции JavaScript доступа к прикладному интерфейсу ASP.NET 2.0 Forms Authentication API.

Глава 9 «Локализация и глобализация приложений» охватывает тему создания веб-приложений, которые могут работать с различными языками и региональными настройками.

Глава 10 «Использование Control Toolkit» представляет собой введение в ASP.NET Control Toolkit – комплект серверных элементов управления, дополненных функциональными возможностями Ajax.

Глава 11 «Добавление анимационных эффектов в веб-страницу» описывает платформу для создания анимационных эффектов, входящую в состав ASP.NET AJAX Control Toolkit.

Глава 12 «Автодополнение ввода пользователя, борьба со спамом и не только» описывает особенности (возможно, спорные) ASP.NET AJAX Control Toolkit, демонстрирует разнообразные инструментальные средства, содержит некоторые советы и рассказывает о наиболее удачных приемах использования.

Глава 13 «Создание собственных элементов управления и помощь сообществу» рассказывает о том, как с помощью платформы Control Toolkit писать собственные элементы управления и как интегрировать их с существующими элементами в проекте.

Глава 14 «Клиентские элементы управления» описывает элементы управления, используемые на стороне клиента, которые поставляются в комплекте с ASP.NET AJAX Futures CTP. Они позволяют упростить обращение к HTML-элементам из сценариев JavaScript с использованием непротиворечивого прикладного интерфейса.

Глава 15 «Привязка и проверка данных» демонстрирует, как реализовать привязку данных к элементам управления на стороне клиента с помощью ASP.NET AJAX Futures CTP.

Глава 16 «Использование аспектов клиентского поведения и компонентов» демонстрирует аспекты поведения (behaviors), встроенные в ASP.NET AJAX, и порядок связывания их функциональности с клиентскими компонентами и элементами управления.

Глава 17 «Использование данных, размещенных на стороне сервера» объясняет, как организовать работу с базами данных. Платформа ASP.NET AJAX может связываться с источниками данных посредством специализированных веб-служб, облегчая доступ к данным без необходимости обновлять всю страницу. Кроме того, в ASP.NET AJAX имеются специализированные элементы управления, предназначенные для отображения данных на стороне клиента.

Глава 18 «Использование внешних веб-служб» поможет преодолеть ограничения общности происхождения, которые накладывает политика безопасности JavaScript, и обеспечить возможность обращения к удаленным веб-службам посредством моста на стороне сервера.

Глава 19 «Использование анимационных эффектов» демонстрирует некоторые возможности ASP.NET AJAX Futures CTP по созданию анимационных эффектов.

Глава 20 «Кнопки Назад/Вперед и закладки» рассказывает о возможных решениях двух наиболее раздражающих проблем, которые присущи Ajax-приложениям (связанных с нарушением стандартного поведения броузера).

Глава 21 «Веб-части» демонстрирует, что можно реализовать с помощью веб-частей (web parts) ASP.NET AJAX и что нельзя, включая, к примеру, реализацию механизма «перетащить-и-отпустить» (drag-and-drop), который будет работать в любом броузере.

Глава 22 «ASP.NET AJAX в комбинации с другими серверными технологиями» наглядно демонстрирует, что некоторые части Microsoft Ajax Library никак не связаны с ASP.NET 2.0. Приводится пример приложения на языке PHP, которое демонстрирует, как можно соединить эти два мира.

Приложение A «Отладка приложений ASP.NET AJAX» рассказывает, как выполняется поиск ошибок в приложениях ASP.NET AJAX и описывает некоторые инструментальные средства броузеров, которые желательно иметь каждому разработчику.

Приложение B «Справочник по XMLHttpRequest» содержит перечень методов и свойств объекта `XMLHttpRequest`.

Приложение C «Справочник DOM» описывает наиболее важные методы JavaScript объектной модели документа (DOM).

Приложение D «Справочник по ASP.NET AJAX» описывает наиболее важные методы, предоставляемые платформой ASP.NET AJAX.

Приложение E «Справочник по ScriptManager, UpdatePanel, UpdateProgress и Timer» документирует свойства этих четырех ключевых серверных элементов управления платформы ASP.NET AJAX.

Технические требования

Для опробования примеров из этой книги необходимо и достаточно иметь платформу ASP.NET 2.0, которая включена в состав свободно распространяемой платформы .NET. Однако, чтобы получить максимум от использования ASP.NET и ASP.NET AJAX, вам необходимо иметь какую-либо интегрированную среду разработки из предлагаемых компанией Microsoft. Среда Visual Web Developer 2005 Express Edition (VWD) распространяется бесплатно. Visual Studio 2005 (любые ее редакции) – пакет программного обеспечения, который обладает богатыми возможностями, распространяется на коммерческой основе. Любой из упомянутых инструментов прекрасно подойдет для опробования примеров из этой книги.

Типографские соглашения

В этой книге приняты следующие соглашения:

Обычный шрифт

Применяется для обозначения пунктов меню, кнопок и комбинаций клавиш (таких как Alt и Ctrl).

Курсив

Обозначает новые термины, адреса электронной почты и веб-сайтов, имена файлов и каталогов, расширений имен файлов, а также названия утилит Unix.

`Моноширинный шрифт`

Применяется для выделения команд, параметров, ключей, переменных, атрибутов, имен функций, типов, классов, пространств имен, методов, модулей, свойств, аргументов, значений, объектов, событий, обработчиков событий, тегов XML, тегов HTML, макроопределений, содержимого файлов и результатов работы команд.

`Моноширинный жирный`

Используется для выделения участков программного кода.

`Моноширинный курсив`

Используется для выделения элементов, которые необходимо заменить на реальные значения.

Этим значком обозначаются советы, предложения и примечания общего характера.

Этим значком обозначаются предупреждения и предостережения.

Использование программного кода примеров

Данная книга призвана оказать вам помощь в решении ваших задач. Вообще вы можете свободно использовать примеры программного кода из этой книги в своих приложениях и в документации. Вам не нужно обращаться в издательство за разрешением, если вы не собираетесь воспроизводить существенные части программного кода. Например, если вы разрабатываете программу и используете в ней несколько отрывков программного кода из книги, вам не нужно обращаться за разрешением. Однако в случае продажи или распространения компакт-дисков с примерами из этой книги вам *необходимо* получить разрешение от издательства O'Reilly. Для цитирования данной книги или примеров из нее, при ответе на вопросы не требуется получение разрешения. При включении существенных объемов программного кода примеров из этой книги в вашу документацию вам *необходимо* будет получить разрешение издательства.

Мы приветствуем, но не требуем добавлять ссылку на первоисточник при цитировании. Под ссылкой на первоисточник мы подразумеваем указание авторов, издательства и ISBN. Например: «Programming ASP.NET AJAX, by Christian Wenz. Copyright 2007 Christian Wenz, 978-0-596-51424-2».

При необходимости использовать значительный объем программного кода примеров из этой книги, за получением разрешения обращайтесь по адресу *permissions@oreilly.com*.

Отзывы и предложения

С вопросами и предложениями, касающимися этой книги, обращайтесь в издательство:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (в Соединенных Штатах Америки или в Канаде)
707-829-0515 (международный)
707-829-0104 (факс)

Список опечаток, файлы с примерами и другую дополнительную информацию вы найдете на сайте книги:

http://www.oreilly.com/catalog/9780596514242

Свои комментарии и вопросы технического характера отправляйте по адресу:

bookquestions@oreilly.com

Дополнительную информацию о книгах, обсуждения, Центр ресурсов издательства O'Reilly вы найдете на сайте:

http://www.oreilly.com

Safari® Books Online

Если на обложке технической книги есть пиктограмма «Safari® Books Online», это означает, что книга доступна в Сети через O'Reilly Network Safari Bookshelf.

Safari предлагает намного лучшее решение, чем электронные книги. Это виртуальная библиотека, позволяющая без труда находить тысячи лучших технических книг, вырезать и вставлять примеры кода, загружать главы и находить быстрые ответы, когда требуется наиболее верная и свежая информация. Она свободно доступна по адресу *http://safari.oreilly.com*.

Благодарности (к книге «Programming Atlas»)

Работа над этой книгой оказалась непростой задачей. Нехватка документации, которая описывала бы изменения от версии к версии, и сложности в отладке программного кода JavaScript обусловили появление большого числа ошибок и необходимость проведения экспериментов. При том, что я работал с ASP.NET и JavaScript в течение длительного времени, Atlas мне пришлось изучать с самого начала. К счастью, разработчики Atlas оказались готовы оказать поддержку и открыты для контактов, в частности, на общедоступном форуме по адресу: *http://forums.asp.net/default.aspx?GroupID=34*.

Я благодарен большому коллективу технических редакторов, которые помогли мне оформить эту книгу и с которыми мы работали в режиме обратной связи. Далее в алфавитном порядке перечислены имена тех, кто помог мне спасти свою репутацию в нескольких случаях: Адонис Битар (Adonis Bitar), Арсен Еремин (Arsen Yeremin), Бертран Ле Рой (Bertrand Le Roy), Кристоф Уилл (Christoph Wille), Майк Поуп (Mike Pope) и Тобиас Хаузер (Tobias Hauser).

Кроме того, я весьма признателен своему редактору Джону Осборну (John Osborn), который руководил этим проектом. Он – единственный известный мне редактор, который постоянно проявлял неудовольствие, когда я отправлял подготовленный материал *до* условленного

крайнего срока. Но именно его превосходное руководство проектом позволило мне сосредоточиться на работе и оставаться при этом в графике (а иногда и опережать его).

Должен признать, что я не большой приверженец выражения личной благодарности членам семьи, мужьям/женам/невестам/партнерам, а также кошкам/собакам. (Единственное исключение – Ричард Хандхаузен (Richard Hundhausen), который когда-то выразил благодарность за то, что в районе, где он жил, не оказалось круглосуточной службы, выполняющей расторжение брака.) Тем не менее я хотел бы воспользоваться такой возможностью и поблагодарить моих родителей. Они очень поддерживали меня в период, когда я работал над своей первой книгой, и теперь, когда у меня за плечами уже более 50 книг, я хочу выразить им свою признательность. Мне неловко признаться, но они иногда умудряются находить ошибки, даже не зная описываемых технологий: однажды мой отец заметил, что в листинге содержится неравное число открывающих и закрывающих скобок. Спасибо вам, Мама и Папа. И еще – спасибо моим друзьям и семье, которые, кажется, относились ко мне с терпением, когда я подолгу работал над книгами или уезжал на еще какую-нибудь конференцию.

Благодарности (к книге «Programming ASP.NET AJAX»)

Иногда оказывается так, что вы просто неудачно выбрали время. Спустя приблизительно две недели после выхода в свет первого издания книги под названием «Programming Atlas» компания Microsoft изменила название платформы на ASP.NET AJAX. Неудачность выбора времени обусловлена не только изменением названия – в Microsoft также весьма радикально изменили внутреннюю архитектуру платформы. Как следствие этого – все примеры программного кода для Atlas оказались неработоспособными в ASP.NET AJAX. Следует признать, что переписать большую часть примеров не составило большого труда, но некоторые функциональные возможности платформы были утрачены или изменены до неузнаваемости.

Вследствие этого данное издание книги полностью отличается от предыдущего. Структура книги претерпела существенные изменения: в нее было добавлено большое число новых глав, какой-то материал был добавлен, какой-то пришлось выбросить, а некоторые главы были полностью переписаны. Таким образом, хотя технически это второе издание – это более или менее новая книга. Однако если у вас имеется программный код, основанный на особенностях платформы Atlas, не волнуйтесь: в нескольких главах этой книги вы найдете рекомендации по переносу устаревшего программного кода на новую версию платформы.

Я выражаю благодарность моему редактору Джону Осборну (John Osborn), который руководил проектом, постоянно высказывая новые

идеи, касающиеся книги. Главным техническим редактором был Майк Поуп (Mike Pope) – эту роль он взял на себя еще при подготовке первого издания книги. Он не только удалял мои остроты в адрес Microsoft (вздох), но и поставлял бесконечное число предложений, комментариев и идей для этого издания. Это был тяжелый труд как для него (поиск ошибок), так и для меня (их исправление), но, на мой взгляд, получившийся результат стоил затраченных усилий. Спасибо вам обоим за ваши усилия, направленные на то, чтобы сделать второе издание книги еще лучше.

Кроме того, спасибо всем читателям первого издания, приславшим огромное число отзывов и предложений. Спасибо всем разработчикам, выучившимся с помощью этой книги, кто также высказывал мне свои замечания.

9

Локализация и глобализация приложений

Я хочу рассказать вам одну поучительную историю…

Когда издатели присылают мне авторские экземпляры моих книг, я обычно раздаю их своим друзьям, разыгрываю их в своем блоге или просто складываю у себя в подвале. Одним словом, я не стремлюсь коллекционировать свои книги. Однако из этого правила есть одно исключение: мне ужасно нравится приобретать их иностранные переводы. Надо сказать, что издатели не всегда получают мои изданные за рубежом книги, и в таких случаях я вовсе не претендую на дарственные экземпляры. Но даже если к издателю и попадает какое-нибудь количество таких переводных изданий, то порой проходит целая вечность, прежде чем они достаются мне. Поэтому, когда до меня доходит информация, что очередная моя книга издана в другой стране, я запускаю броузер и начинаю «охоту».

Обычно поиски приводят меня на сайт какого-нибудь книжного Интернет-магазина, где вся информация выводится на незнакомом для меня языке, и потому с большими трудностями мне удается вписать свой адрес и номер кредитной карты в соответствующие поля. Я считаю себя счастливчиком, потому что мои анкетные данные пока еще не были перехвачены злоумышленниками (у меня есть специальная кредитная карта для таких «серых заказов», как я их называю). В результате всего этого время от времени я получаю посылки, которые совершают чуть ли не кругосветное путешествие, прежде чем попадают ко мне.

Но почему, по какой причине я должен испытывать подобные сложности? В век глобализации владельцы веб-сайтов могут существенно расширить круг своих посетителей, если будут общаться на одном с ними языке. На мой взгляд, есть две основные причины, почему основная масса веб-сайтов остаются одноязычными. Первая: перевод сайтов –

удовольствие дорогостоящее, а в зависимости от целевой аудитории, часто выгода от наличия иноязычной версии сайта не покрывает затрат на ее поддержку. Вторая: препятствия технического характера. Если при создании многоязычного сайта вы хотите избежать использования «дизайн-метода», основанного на операциях копирования и вставки, вам потребуется некоторый механизм, который позволит выполнять перевод с минимальными усилиями. Благодаря растущей значимости и быстрому распространению приложений, построенных на базе JavaScript и Ajax, реализация перевода также становится все более значимой и востребованной.

Как всегда, можно выработать свое собственное решение этой задачи, но платформа ASP.NET AJAX уже имеет некоторую поддержку локализации и глобализации. Это позволяет создавать многоязычные версии веб-сайтов, обладающие поддержкой ASP.NET AJAX, и создавать сайты, которые могут предоставлять пользователю ту или иную свою версию, в зависимости от региональных настроек броузера. (Если раньше вы не знали, что броузер может передавать серверу список предпочитаемых языков, вы сможете прочитать об этом ниже, в этой же главе.)

Локализация

Локализация веб-сайта – это процесс адаптации содержимого к региональным настройкам системы, чаще всего – к региональным настройкам пользователя. Часто локализация как таковая обозначается аббревиатурой l10n, которая происходит от обозначения «*l*, затем 10 символов, затем *n*»[1] (так называемый *нумероним*).

На любом веб-сайте имеются самые разные разделы, которые могли бы быть локализованы. И сам текст на сайте, и название валюты, и формат представления даты и времени – все это может быть локализовано. В платформе ASP.NET имеется ряд возможностей по обеспечению локализации (некоторые ссылки вы найдете в разделе «Для дополнительного чтения» в конце главы). Некоторые из этих возможностей используются платформой ASP.NET AJAX для организации поддержки локализации в программном коде JavaScript.

Локализация сценариев

Самый простой способ локализации состоит в том, чтобы написать специализированный сценарий, который будет решать поставленную задачу. Например, этот сценарий мог бы определять региональные настройки и затем загружать ту или иную библиотеку. Половину этой задачи можно решить средствами ASP.NET, а вторую половину – средствами ASP.NET AJAX.

---

[1] Слово «локализация» в английском языке записывается как localization, то есть «l», затем 10 символов («ocalizatio»), затем символ «n». – *Прим. перев.*

Ниже приводится пример короткого сценария JavaScript, который выводит текущую дату в локализованном формате. В файле JavaScript *Dayname.js* определяются две переменные:

`dateformat`

Строка с форматом представления даты, где используются следующие заполнители: `ss` (день недели), `dd` (число месяца), `mm` (месяц) и `yyyy` (год)

`daynames`

Массив с локализованными названиями всех семи дней недели

В примере 9.1 приводится содержимое файла *Dayname.js*.

Пример 9.1. Информация о дате, локализованная для английского языка

```
Dayname.js

var dateformat = "ss, yyyy-mm-dd";

var daynames = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
                "Friday", "Saturday"];
```

Перевести содержимое этого файла на другой язык не представляет труда. В примере 9.2 приводится тот же самый файл, но на этот раз вся информация переведена на немецкий язык. В обоих этих языках используются различные форматы представления дат и названия дней недели.

Пример 9.2. Информация о дате, локализованная для немецкого языка

```
Dayname.de-DE.js

var dateformat = "ss, dd.mm.yyyy";

var daynames = ["Sonntag", "Montag", "Dienstag", "Mittwoch", "Donnerstag",
                "Freitag", "Samstag"];
```

Для локализованных версий сценариев очень важную роль играют имена их файлов. Непосредственно перед расширением файла (*.js*) вставляется информация о языке и, возможно, о регионе, в формате язык-регион, где язык обозначается двухсимвольным кодом языка, а регион обозначается двухсимвольным кодом региона. Код языка, само собой разумеется, определяет язык, на котором написан текст. Код региона определяет форматы представления тех или иных данных (в различных регионах или странах могут говорить на одном и том же языке, но использовать разные соглашения по форматированию – как в случае Соединенных Штатов и Великобритании). Например, американский английский обозначается как en-US, британский английский – en-UK, немецкий (Германия) – de-DE, немецкий (Австрия) – de-AT и так далее. Коды определяются международной организацией по стандартизации ISO, и, в соответствии с соглашениями, код языка записывается строчными символами, а код региона – прописными. В примере 9.2 файл сценария имеет имя *Dayname.de-DE.js*, указывая на немецкий язык и Германию.

Теперь создадим новую страницу *.aspx*, которая будет использовать эти локализованные файлы. В начале страницы расположим элемент `<span>`, который будет использоваться для динамического вывода локализованной даты. Обратите внимание на символ неразрывного пробела (` `) внутри элемента `<span>` – он очень важен для программного кода JavaScript. Обычный пробел не даст желаемого эффекта из-за особенностей поведения Internet Explorer.

```
<span id="date"> </span>
```

Теперь можно добавить программный код JavaScript, который будет использовать переменные `dateformat` и `daynames`, определения которых находятся во внешних файлах JavaScript для вывода даты в локализованном формате:

```
<script type="text/javascript">
  function pageLoad() {
    var d = new Date();
    var datestring = dateformat.replace("ss", daynames[d.getDay()])
                               .replace("dd", d.getDate())
                               .replace("mm", d.getMonth() + 1)
                               .replace("yyyy", d.getFullYear());
    $get("date").firstChild.nodeValue = datestring;
  }
</script>
```

Пока что нет ничего необычного. Но дальше ASP.NET и ASP.NET AJAX начинают свои магические манипуляции. Первое: платформа ASP.NET AJAX должна загрузить внешний файл JavaScript и она (это просто чудо) отыскивает файл, соответствующий текущим языку и региону, основываясь на имени файла. Для загрузки файла *Dayname.js* внутри элемента управления `ScriptManager` используется элемент `<Scripts>`. Основную функцию на этом этапе выполняют два атрибута:

`ResourceUICultures` (*атрибут элемента* `<asp:ScriptReference>`)

Содержит перечень всех поддерживаемых культур (разделенных запятыми), для которых существует перевод.

`EnableScriptLocalization` (*атрибут элемента* `<asp:ScriptManager>`)

Если этот атрибут установить в значение `true`, для текущей страницы будет активирована поддержка механизма локализации ASP.NET AJAX:

```
<asp:ScriptManager ID="ScriptManager1" runat="server"
  EnableScriptLocalization="true">
  <Scripts>
    <asp:ScriptReference Path="Dayname.js"
                                    ResourceUICultures="de-DE,fr-FR" />
  </Scripts>
</asp:ScriptManager>
```

По умолчанию используется файл *Dayname.js*, однако если будет использована одна из культур, перечисленных в атрибуте `ResourceUICul-`

tures (то есть если броузер запросит одну из культур, основываясь на своих настройках), то будет загружен соответствующий локализованный файл.

До сих пор страница еще не установила свою культуру в соответствии с региональными настройками броузера клиента. В этом месте на сцене появляется ASP.NET. Следующая ниже директива Page автоматически установит значение культуры в соответствии с настройками броузера:

```
<%@ Page Language="C#" UICulture="auto" %>
```

Броузер, настроенный на предпочтительное использование французского языка, загрузит файл *Dayname.fr-FR.js*, с другой стороны, броузер, настроенный на предпочтительное использование итальянского языка, загрузит файл *Dayname.js*, так как для итальянского языка нет локализованного сценария. То есть файл *Dayname.js* используется по умолчанию в случае отсутствия локализации для данного языка.

В примере 9.3 приводится полный программный код страницы, которая использует механизм локализации ASP.NET AJAX, который мы только что рассмотрели. Для опробования примера вам необходимо переписать файл *Dayname.js* в корневой каталог вашего веб-сайта.

На рис. 9.1 показано, как выглядит страница в броузере, настроенном на использование английского языка. На рис. 9.2 видно, что в броузере, настроенном на использование немецкого языка, дата отображается в формате, типичном для Германии.

Пример 9.3. Локализация сценария

```
Localization-Inline.aspx

<%@ Page Language="C#" UICulture="auto" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">
</script>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>ASP.NET AJAX</title>
  <script type="text/javascript">
    function pageLoad() {
      var d = new Date();
      var datestring = dateformat.replace("ss", daynames[d.getDay()])
                                 .replace("dd", d.getDate())
                                 .replace("mm", d.getMonth() + 1)
                                 .replace("yyyy", d.getFullYear());
      $get("date").firstChild.nodeValue = datestring;
    }
  </script>
</head>
<body>
```

Рис. 9.1. *Страница на английском языке*

Рис. 9.2. *Страница на немецком языке*

```
    <form id="form1" runat="server">
      <asp:ScriptManager ID="ScriptManager1" runat="server"
                         EnableScriptLocalization="true">
        <Scripts>
          <asp:ScriptReference Path="Dayname.js"
                               ResourceUICultures="de-DE,fr-FR" />
        </Scripts>
      </asp:ScriptManager>
      <div>
        <span id="date"> </span>
      </div>
    </form>
  </body>
  </html>
```

Использование сопутствующих ресурсов из ASP.NET AJAX

Другой способ локализации приложений ASP.NET заключается в использовании так называемых *сопутствующих ресурсов* (иногда их называют *сопутствующими сборками*). Под ними подразумеваются скомпилированные внешние файлы ресурсов, которые загружаются, только

если это необходимо для текущей культуры. Платформа ASP.NET AJAX допускает возможность добавления файлов JavaScript (*.js*) в виде ресурсов в сопутствующие сборки, а платформа Ajax обеспечивает механизм использования данных из программного кода JavaScript.

В примере 9.4 используются возможности ASP.NET AJAX по локализации, заключенные в компонент ASP.NET, который может использоваться в веб-странице ASP.NET. На примере этого компонента иллюстрируется решение двух задач. Во-первых, загрузка корректной сопутствующей сборки с помощью ASP.NET AJAX. Во-вторых, доступ к корректным локализованным данным для вывода текста на языке пользователя броузера из сценария, размещенного в сборке.

Для создания компонентов необходимо иметь Visual Studio 2005, так как потребуется создавать скомпилированные сборки. Если вы пользуетесь Visual Web developer Express Edition, то эта среда разработки не поддерживает создание сборок. Однако вы можете установить Visual C# Express Edition (еще один бесплатный инструмент, доступный для загрузки) и пользоваться им.

Для начала создайте новый проект (рис. 9.3). Дайте проекту имя `LocalizedDate`. Вообще это имя не является обязательным, но оно будет использоваться повсюду в этом примере, и поэтому вам предлагается использовать это имя, чтобы следовать за дальнейшим описанием.

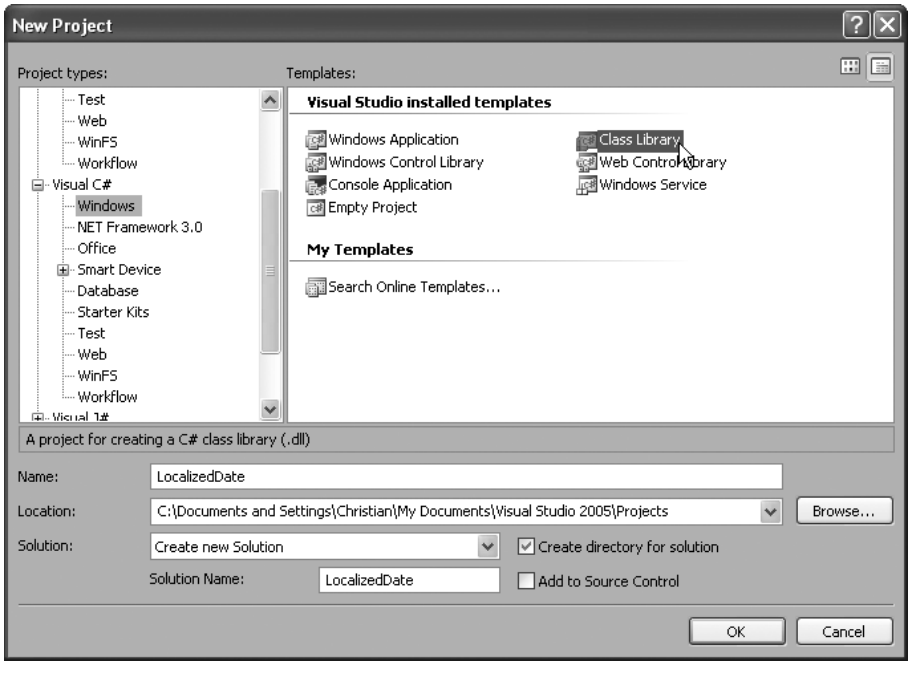

Рис. 9.3. *Создание нового проекта библиотеки классов*

Если вы пользуетесь Visual C# Express Edition, вам придется создавать новый проект с нуля. Пользователи Visual Studio 2005 могут просто добавить новый проект к существующему веб-сайту ASP.NET AJAX. В любом случае это должен быть проект библиотеки классов, как показано на рис. 9.3.

Прежде всего, в проект необходимо добавить ссылки на `System.Web` и `System.Web.Extensions`. Затем добавить некоторые файлы ресурсов. Как показано на рис. 9.4, мы начнем с файла ресурсов английской локализации *DateResources.resx*, который одновременно будет использоваться как исходный. Фактически на рис. 9.4 показан лишь инструмент с графическим интерфейсом для редактирования файла *DateResources.resx*. Внутри файлы ресурсов содержат текст в формате XML. В примере 9.4 показана разметка XML для файла ресурсов с немецкой локализацией. Как можно заметить, применение инструмента с графическим интерфейсом облегчает работу и снижает вероятность появления ошибок. (Чтобы перейти в режим редактирования файла ресурсов в формате XML, перейдите в окно Visual Studio Solution Explorer (обозреватель решений), щелкните правой кнопкой мыши на файле и выберите пункт контекстного меню View Code (просмотр кода).) Создайте немецкую версию файла ресурсов и дайте ему имя *DateResources.de.resx*.

Пример 9.4. Немецкий файл ресурсов

`DateResources.de.resx`

```
<?xml version="1.0" encoding="utf-8"?>
<root>
  <xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

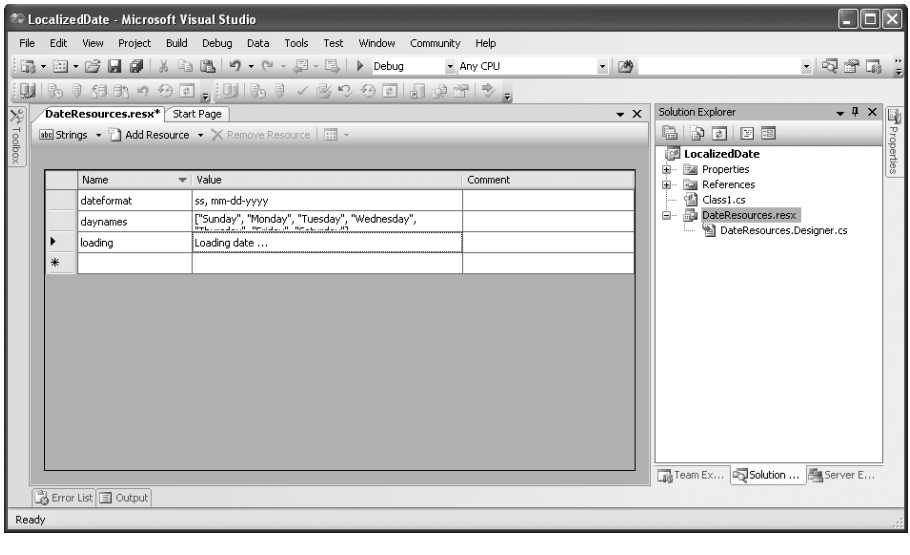

***Рис. 9.4.** Английский файл ресурсов*

```
          xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
<xsd:import namespace="http://www.w3.org/XML/1998/namespace" />
<xsd:element name="root" msdata:IsDataSet="true">
  <xsd:complexType>
    <xsd:choice maxOccurs="unbounded">
      <xsd:element name="metadata">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="value" type="xsd:string" minOccurs="0" />
          </xsd:sequence>
          <xsd:attribute name="name" use="required" type="xsd:string" />
          <xsd:attribute name="type" type="xsd:string" />
          <xsd:attribute name="mimetype" type="xsd:string" />
          <xsd:attribute ref="xml:space" />
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="assembly">
        <xsd:complexType>
          <xsd:attribute name="alias" type="xsd:string" />
          <xsd:attribute name="name" type="xsd:string" />
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="data">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="value" type="xsd:string" minOccurs="0"
                msdata:Ordinal="1" />
            <xsd:element name="comment" type="xsd:string" minOccurs="0"
                msdata:Ordinal="2" />
```

Установка культуры в ASP.NET

В примере 9.1 приводится вариант автоматического определения используемой культуры. Однако платформа ASP.NET AJAX обеспечивает механизм динамического изменения культуры. Вы можете дать пользователю возможность изменять культуру страницы, например, с помощью элемента `LinkButton` или раскрывающегося списка. (Нередко используются национальные флаги, но такую практику нельзя признать удачной, поскольку на одном и том же языке могут говорить в нескольких странах, а в некоторых странах официальными признаны несколько языков.)

Для изменения культуры текущей страницы используется свойство `System.Threading.Thread.CurrentThread.CurrentUICulture`. В пространстве имен `System.Globalization` существуют вспомогательные методы создания подходящей культуры для свойства `CurrentUICulture` в текущем потоке. Дополнительная информация об этой особенности ASP.NET приводится в разделе «Для дополнительного чтения» в конце главы.

```
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required"
                 msdata:Ordinal="1" />
            <xsd:attribute name="type" type="xsd:string"
                 msdata:Ordinal="3" />
            <xsd:attribute name="mimetype" type="xsd:string"
                 msdata:Ordinal="4" />
            <xsd:attribute ref="xml:space" />
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="resheader">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="value" type="xsd:string" minOccurs="0"
                   msdata:Ordinal="1" />
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required" />
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
<resheader name="resmimetype">
  <value>text/microsoft-resx</value>
</resheader>
<resheader name="version">
  <value>2.0</value>
</resheader>
<resheader name="reader">
  <value>System.Resources.ResXResourceReader, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
</resheader>
<resheader name="writer">
  <value>System.Resources.ResXResourceWriter, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
</resheader>
<data name="dateformat" xml:space="preserve">
  <value>ss, dd.mm.yyyy</value>
</data>
<data name="daynames" xml:space="preserve">
  <value>["Sonntag", "Montag", "Dienstag", "Mittwoch", "Donnerstag",
          "Freitag", "Samstag"]</value>
</data>
<data name="loading" xml:space="preserve">
  <value>Lade Datum ...</value>
</data>
</root>
```

Из этого XML-файла видно, какая информация помещается в файлы ресурсов:

`dateformat`

Формат представления даты с использованием символов-заполнителей

`daynames`

Локализованные названия дней недели

`loading`

Текст «Loading date...» (загрузка даты), переведенный на отдельные языки

Вам необходимо создать как минимум два файла ресурсов (для английского и немецкого языков). По своему желанию вы можете добавить локализации и для других языков. Для определения названий дней недели используйте синтаксис массивов в формате JSON (глава 3). Будьте внимательны и старайтесь не допускать опечаток, так как приложения, состоящие из серверных и клиентских сценариев, а также ресурсов, очень сложно отлаживать.

В окне Solution Explorer (обозреватель решений) в текущем проекте откройте папку *Properties* (свойства). Внутри нее вы найдете файл *AssemblyInfo.cs*, в котором содержится дополнительная информация о проекте. Если вы не видите этот файл, щелкните на кнопке Show All Files (показать все файлы), как показано на рис. 9.5.

В конец файла *AssemblyInfo.cs* добавьте следующие строки:

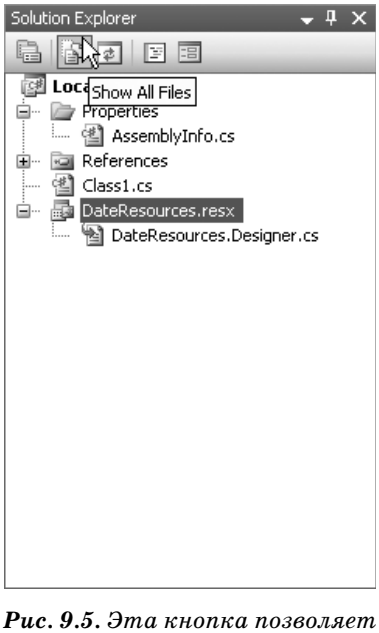

Рис. 9.5. *Эта кнопка позволяет увидеть файл AssemblyInfo.cs*

```
[assembly: System.Web.UI.WebResource("LocalizedDate.Dayname.js",
"application/xjavascript")]
[assembly: System.Web.UI.ScriptResource("LocalizedDate.Dayname.js",
"LocalizedDate.DateResources", "LocData")]
```

Обратите внимание на значение `LocData` в последнем аргументе. Это имя позволит обращаться ко всем данным в файле ресурсов из программного кода JavaScript посредством объекта `LocData`. Имя может выбираться совершенно произвольно, но это имя будет использоваться в следующих примерах.

Остальные значения в предыдущем фрагменте содержат данные, имеющие отношение к приложению. `LocalizedDate` – это имя проекта, *Dayname.js* – это файл JavaScript, который будет создан на следующем шаге, а `DateResources` будет использоваться позже для организации доступа к данным в файле ресурсов (из программного кода на C#).

Файл *Dayname.js* содержит программный код JavaScript, который определяет текущую дату и форматирует ее в соответствии с текущими региональными настройками. Как уже говорилось выше, объект `LocData` используется в JavaScript для доступа к информации из файла ресурсов. Свойство `LocData.dateformat` содержит строку формата даты, а свойство `LocData.daynames` – массив с названиями дней недели. Последнее значение хранится в формате JSON (глава 3), поэтому нам придется использовать функцию `eval()` для преобразования строки в объект JavaScript:

```
var daynames = eval("(" + LocData.daynames + ")");
```

Программный код в *Dayname.js*, подобный тому, что приводился в примере 9.3, будет замещать символы-заполнители в строке `dateformat` и отображать результат в окне броузера. Обратите внимание на односекундную задержку, которая была добавлена для того, чтобы сделать эффект более наглядным. В примере 9.5 приводится полный программный код файла JavaScript. Добавьте его в свой проект.

Пример 9.5. Программный код JavaScript в библиотеке классов

Dayname.js

```
Sys.Application.add_load(function() {
  var d = new Date();
  var daynames = eval("(" + LocData.daynames + ")");
  var datestring = LocData.dateformat.replace("ss", daynames[d.getDay()])
                                     .replace("dd", d.getDate())
                                     .replace("mm", d.getMonth() + 1)
                                     .replace("yyyy", d.getFullYear());
  setTimeout('$get("date").firstChild.nodeValue = "'
             + datestring + '"', 1000);
});
```

В заключение файл *Dayname.js* необходимо встроить в сборку. Щелкните на имени файла в окне Solution Explorer (обозреватель решений).

В окне Properties (свойства) установите параметр Build Action (действие по сборке) в значение Embedded Resource (встроенный ресурс), как показано на рис. 9.6. Это приведет к тому, что во время компиляции проекта файл JavaScript будет внедрен непосредственно в файл DLL библиотеки.

В программном коде из *Dayname.js* вы можете заметить вызов функции `$get()` для доступа к еще не определенному элементу с именем `"date"`. Этот элемент будет объявлен как часть управляющего кода в файле с классом на C#.

В окне Solution Explorer (обозреватель решений) переименуйте файл *Class1.cs* в *LocDateControl.cs* (для этого нужно щелкнуть правой кнопкой мыши на имени файла и выбрать в контекстном меню пункт Rename (переименовать)) и выберите вариант обновления всех ссылок. В этом файле будет располагаться программный код компонента ASP.NET на языке C#. Этот компонент будет наследовать свойства и методы класса `Sys.Web.UI.Control` платформы .NET.

В классе `LocDateControl` мы переопределим метод `CreateChildControls()` базового класса и добавим свой собственный HTML-элемент – метку `<span>` с именем `"date"`:

```
hgc = new HtmlGenericControl();
hgc.TagName = "span";
hgc.ID = "date";
```

В элементе `<span>` будет находиться текст «Loading date...» или тот, который задан для выбранного языка. Чтобы получить нужную строку,

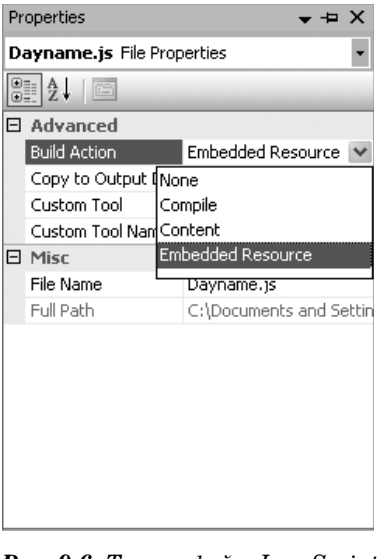

Рис. 9.6. *Теперь файл JavaScript будет встроен в файл библиотеки*

компонент должен обратиться к встроенным ресурсам. Эту операцию выполняет следующий фрагмент:

```
System.Resources.ResourceManager r = new System.Resources.ResourceManager(
  "LocalizedDate.DateResources",
  this.GetType().Assembly);
hgc.InnerHtml = r.GetString("loading");
```

И наконец, новый компонент с локализованным текстом добавляется на страницу:

```
Controls.Add(hgc);
```

В примере 9.6 приводится полный программный код для файла библиотеки классов.

Пример 9.6. Программный код для библиотеки классов

LocDateControl.cs

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Web.UI;
using System.Web.UI.HtmlControls;

namespace LocalizedDate
{
  public class LocDateControl : Control
  {
    private HtmlGenericControl hgc;

    protected override void CreateChildControls()
    {
      base.CreateChildControls();

      hgc = new HtmlGenericControl();
      hgc.TagName = "span";
      hgc.ID = "date";

      System.Resources.ResourceManager r = new
      System.Resources.ResourceManager(
          "LocalizedDate.DateResources",
          this.GetType().Assembly);
      hgc.InnerHtml = r.GetString("loading");

      Controls.Add(hgc);
    }
  }
}
```

Теперь можно выполнить сборку проекта библиотеки классов. В результате будут созданы два файла. Файл *LocalizedDate.dll*, содержащий реализацию компонента в библиотеке классов, и файл сборки, содержащий файл ресурсов по умолчанию (из *DateResources.resx*). Для

каждого дополнительного языка будет создаваться каталог с сопутствующей сборкой. Например, немецкий перевод будет находиться в каталоге *de*, содержащем сборку *LocalizedDate.resources.dll*.

При использовании Visual Studio 2005 новые сборки, как правило, сразу становятся доступны в приложениях ASP.NET AJAX (если это не так, добавьте явно ссылку на библиотеку классов в проекте веб-сайта). Если вы используете Visual Web Developer, создайте каталог *Bin* в каталоге с приложением ASP.NET AJAX, а затем поместите в этот каталог копии обоих файлов *LocalizedDate.dll* и всех подкаталогов (*de*, ...).

Теперь можно импортировать компонент в любую страницу ASP.NET с помощью следующей директивы:

```
<%@ Register TagPrefix="OReilly" Assembly="LocalizedDate" Namespace="
LocalizedDate" %>
```

Поскольку этот компонент не объявляет никаких общедоступных свойств, он может быть включен в страницу с минимальными усилиями:

```
<OReilly:LocDateControl ID="ldc1" runat="server" />
```

Компонент содержит весь программный код JavaScript, необходимый для отображения экранной заставки («Loading data...») и локализованной даты. Все, что осталось сделать для ASP.NET AJAX, – это загрузить сборку в элемент управления `ScriptManager`. В элементе ScriptReference должны быть указаны оба имени – имя сборки (`localizedDate`) и каноническое имя встроенного файла JavaScript (`LocalizedDate.Dayname.js`). Не забудьте установить атрибут `EnableScriptLocalization` элемента `ScriptManager`!

```
<asp:ScriptManager ID="ScriptManager1" runat="server"
                   EnableScriptLocalization="true">
  <Scripts>
    <asp:ScriptReference Assembly="LocalizedDate"
                         Name="LocalizedDate.Dayname.js" />
  </Scripts>
</asp:ScriptManager>
```

Наконец, настроим страницу так, чтобы она автоматически определяла корректную культуру:

```
<%@ Page Language="C#" UICulture="auto" %>
```

В примере 9.7 приводится полный код страницы *.aspx*. На рис. 9.7 и 9.8 показаны различные состояния приложения.

Пример 9.7. Использование сопутствующей сборки в ASP.NET AJAX

```
Localization-Satellite.aspx

<%@ Page Language="C#" UICulture="auto" %>

<%@ Register TagPrefix="OReilly" Assembly="LocalizedDate" Namespace="
LocalizedDate" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">
</script>

<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
  <title>ASP.NET AJAX</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server"
                       EnableScriptLocalization="true">
      <Scripts>
        <asp:ScriptReference Assembly="LocalizedDate"
                             Name="LocalizedDate.Dayname.js" />
      </Scripts>
    </asp:ScriptManager>
    <div>
      <OReilly:LocDateControl ID="ldc1" runat="server" />
    </div>
  </form>
```

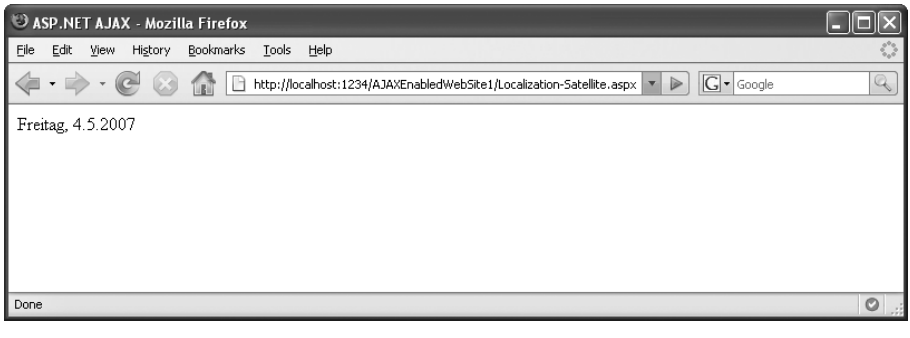

Рис. 9.7. *Страница, локализованная для немецкого языка*

Рис. 9.8. *Оригинальная англоязычная страница*

```
</body>
</html>
```

Глобализация и интернационализация

В дополнение к поддержке локализации, ASP.NET AJAX предоставляет поддержку глобализации, которую иногда еще называют как интернационализация (или i18n, если вам нравится использовать нумеронимы). В элементе управления `ScriptManager` имеется свойство `EnableScriptGlobalization`. Если в этом свойстве установить значение `true`, ASP.NET AJAX будет в состоянии локализовать значения дат. Для этого библиотека Ajax расширяет возможности объекта `Date` из JavaScript (и других объектов – подробности в главе 4 и в приложении D), благодаря чему в этом объекте появился новый метод с именем `localeFormat()`. Этот метод форматирует значение даты в соответствии с региональными настройками в броузере. Эти настройки передаются в HTTP-заголовке `Accept-Language`. На рис. 9.9 показаны HTTP-заголовки, отправляемые броузером Firefox с настройками для немецкого языка. Обратите внимание: заголовок `Accept-Language` включает перечень поддерживаемых языков в сответствии с уровнем их предпочтения. На этой иллюстрации видно, что немецкий язык является наиболее предпочтительным, вслед за ним идет английский язык. В англоязычных броузерах обычно предпочтительным является только английский язык, иногда учитываются различия между американским,

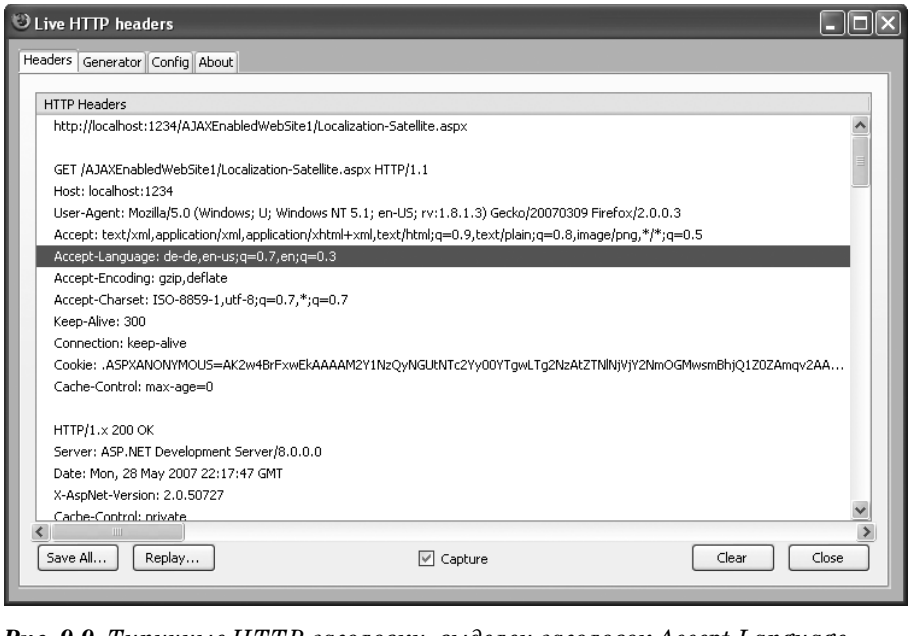

Рис. 9.9. *Типичные HTTP-заголовки, выделен заголовок Accept-Language*

британским и канадским английским. Разумеется, название языка, которое отправляется броузером, не всегда соответствует действительности, поэтому наилучшим вариантом остается возможность явного выбора языка пользователем.

Предпочтительный язык броузера может быть изменен пользователем. В Firefox для этого нужно выбрать пункт меню Правка→Настройки (Tools→Options), в открывшемся диалоге выбрать вкладку Дополнительно (Advanced) и щелкнуть на кнопке Выбрать (Choose). В диалоге Languages (Языки) (рис. 9.10) можно изменить порядок предпочтений, добавить дополнительные языки или удалить существующие.

В Internet Explorer нужно выбрать пункт меню Сервис→Свойства обозревателя (Tools→Internet Options) и в открывшемся диалоге на вкладке Общие (General) щелкнуть на кнопке Языки (Language), в результате откроется диалог, как показано на рис. 9.11.

В других броузерах также имеются похожие средства изменения языковых настроек.

Но вернемся к теме ASP.NET AJAX и глобализации значений дат. Метод `Date.localFormat()` замещает символы-заполнители локализованными названиями дней недели и месяцев. Следующий фрагмент выведет нечто похожее на «Wednesday, 1. May 2007» в зависимости от текущих языковых настроек броузера:

```
<script type="text/javascript">
  function pageLoad() {
    $get("date").firstChild.nodeValue =
      (new Date()).localeFormat("dddd, dd. MMMM yyyy");
  }
</script>
```

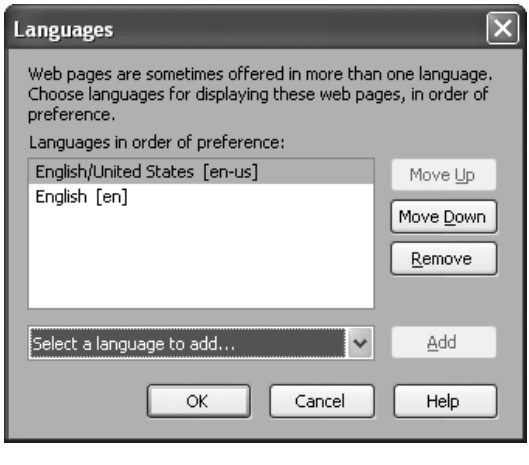

Рис. 9.10. *Изменение языковых предпочтений в броузере Firefox*

Необходимо выполнить еще один шаг. Приложение ASP.NET должно правильно установить культуру. Культура может быть установлена в файле *Web.config*, программно или с помощью разметки в странице. Ссылки на дополнительную информацию о возможных вариантах вы найдете в разделе «Для дополнительного чтения» в конце этой главы.

В следующем примере культура устанавливается с помощью директивы `@ Page`. Мы могли бы установить какую-то конкретную культуру, но предпочитаем доверить платформе ASP.NET AJAX определять корректные настройки из HTTP-заголовка `Accept-Language`. Делается это с помощью следующего объявления:

```
<%@ Page Language="C#" Culture="auto" %>
```

В примере 9.8, который на этот раз получился достаточно коротким, приводится полный код, который делает все что нужно.

Пример 9.8. Глобализация даты

Globalization.aspx

```
<%@ Page Language="C#" Culture="auto" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
```

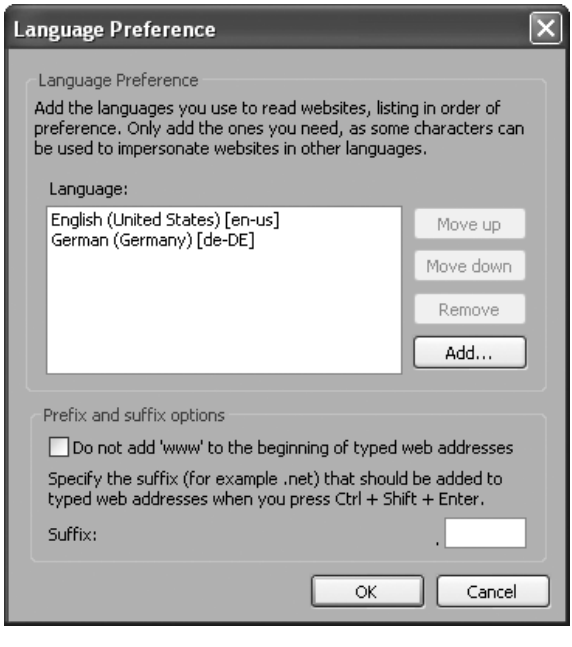

Рис. 9.11. *Изменение языковых предпочтений в Internet Explorer*

```
  <title>ASP.NET AJAX</title>
  <script type="text/javascript">
  function pageLoad() {
    $get("date").firstChild.nodeValue =
      (new Date()).localeFormat("dddd, dd. MMMM yyyy");
  }
  </script>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server"
                       EnableScriptGlobalization="true">
    </asp:ScriptManager>
    <div>
      <span id="date"> </span>
    </div>
  </form>
</body>
</html>
```

На рис. 9.12 показаны результаты работы этой страницы на немецком языке, а на рис. 9.13 – на французском. В обоих случаях эти результаты

Рис. 9.12. *Дата на немецком языке*

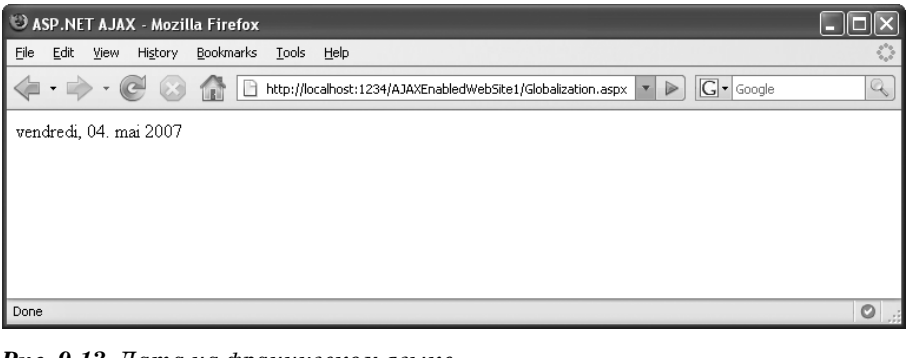

Рис. 9.13. *Дата на французском языке*

были получены простым изменением языковых настроек в броузере. На рисунках видно, что названия дня недели (символ-заполнитель `dddd`) и месяца (`MMMM`) корректно переведены на выбранный язык. Однако формат представления даты локализован не полностью: точка после числа месяца – обычное дело в немецком языке, но не во французском. Для устранения подобных проблем необходимо вручную выполнять локализацию информации о дате, как это было продемонстрировано в примерах 9.3 и 9.7.

Подведение итогов

В век глобализации веб-сайты должны обеспечивать возможность взаимодействия с пользователями из разных стран, которые говорят на разных языках. В этой главе были продемонстрированы некоторые приемы создания многоязычных веб-сайтов с использованием ASP.NET AJAX.

Для дополнительного чтения

http://msdn2.microsoft.com/en-us/library/76091f86-f967-4687-a40f-de87bd8cc9a0.aspx

Информация в MSDN о настройке языковых параметров из ASP.NET

http://ajax.asp.net/docs/tutorials/GlobalizingDateUsingClientScript.aspx

Руководство по глобализации приложений в документации компании Microsoft к ASP.NET AJAX