

Введение

Сразу после появления платформа .NET породила небольшую лавину новых технологий. Это был совершенно новый способ написания Web-приложений (ASP.NET), совершенно новый способ подключения к базам данных (ADO.NET), новые языки программирования с безопасностью в отношении типов (C# и VB.NET) и управляемая исполняющая среда (CLR). Не менее важной среди этих новшеств является Windows Forms — библиотека классов, необходимых для создания Windows-приложений.

Несмотря на то что Windows Forms является зрелым и полнофункциональным инструментальным средством, оно жестко связано с основными конструктивными особенностями Windows, которые не меняются на протяжении последних десяти лет. Более того, Windows Forms основывается на интерфейсе Windows API при создании внешнего вида стандартных элементов пользовательского интерфейса, таких как кнопки, текстовые окна, флажки и т.п. Как результат, эти ингредиенты, по сути, не поддаются настройке. Например, если вы хотите создать элегантную кнопку, вам нужно создать специальный элемент управления и раскрасить каждую частицу кнопки (во всех ее разных состояниях) с помощью низкоуровневой модели рисования. И не думайте даже об анимационных эффектах, таких как вращающийся текст, мерцающие окна или живые окна предварительного просмотра, поскольку вам придется раскрашивать каждую деталь вручную!

Все поменялось благодаря новой модели с совершенно другой структурой, которую предлагает Windows Presentation Foundation (WPF). Несмотря на то что WPF включает уже знакомые вам стандартные элементы управления, она сама рисует каждый текст, рамку и фон. Как результат, WPF может предложить гораздо больше мощных функций, которые помогут вам изменить любой элемент содержимого, визуализируемого на экране. С помощью этих функций вы можете изменить стиль обычных элементов управления, таких как кнопки, зачастую без переписывания какого-либо кода. Точно так же, вы можете использовать объекты трансформации, чтобы вращать, растягивать, изменять масштаб и искажать все, что относится к пользовательскому интерфейсу; вы можете даже использовать встроенную систему анимации в WPF, чтобы все это делалось на глазах у пользователя.

В основе новых возможностей WPF лежит мощная новая инфраструктура, основанная на DirectX — API-интерфейсе аппаратно-ускоренной графики, который обычно используется в современных компьютерных играх. Это означает, что вы можете использовать богатые графические эффекты без ущерба для производительности, как это было бы при использовании Windows Forms. В действительности, вы можете даже получить расширенные функции, такие как поддержка файлов видео и трехмерного содержимого. С их помощью (а также при наличии хорошего инструмента для проектирования) можно создавать потрясающие пользовательские интерфейсы и визуальные эффекты, чего невозможно сделать с помощью Windows Forms.

Несмотря на то что современные функции видео, анимации и трехмерных изображений часто становятся объектом наибольшего внимания в WPF, важно отметить, что вы можете применять WPF и для создания обычных Windows-приложений со стандартными элементами управления и простым внешним видом. В действительности, совсем несложно использовать обычные элементы управления в WPF, как и в Windows Forms. Более того, WPF улучшает функции, которые будут представлять интерес для разработчиков бизнес-приложений, включая существенно улучшенную модель привязки данных, новый набор классов для печати содержимого и управления очередью печати, а также возможность использования документов для отображения больших объемов форматированного текста. Вы получите даже новую модель для создания страничных приложений, плавно выполняющихся в Internet Explorer, и которые могут запускаться с Web-сайта, и все это без обычных предупреждений о безопасности и раздражающих подсказок по установке.

Наконец, WPF комбинирует лучшие качества из старого мира разработки приложений для Windows и новые инновационные технологии для создания современных, насыщенных качественной графикой пользовательских интерфейсов. Несмотря на то что приложения, созданные с помощью Windows Forms, будут существовать еще многие годы, разработчикам, интересующимся новыми проектами разработки приложений для Windows, следует глубже знакомиться с WPF.

На кого рассчитана эта книга

Данная книга предлагает глубокие исследования WPF для профессиональных разработчиков, которые знакомы с платформой .NET, языком программирования C# и средой разработки Visual Studio. Наличие опыта работы с Windows Forms приветствуется, хотя и не является обязательным для освоения материала.

Эта книга дает полное описание каждой главной функциональной особенности WPF, от XAML (язык разметки, применяемый для описания пользовательских интерфейсов WPF) до трехмерной графики и анимации. Вместе с тем, вам будут предложены примеры кода, включающие другие возможности .NET Framework, такие как классы ADO.NET, служащие для реализации запросов к базам данных. Эти возможности здесь не рассматриваются. Если нужны дополнительные сведения о средствах .NET, не являющихся специфическими для WPF, обратитесь к специализированным книгам по .NET.

Как организована эта книга

Эта книга разбита на 26 глав. Если вы только начали знакомство с WPF, вам лучше читать книгу с самого начала, так как в последних главах часто будут упоминаться технологии, описанные в более ранних главах.

Ниже приведен краткий обзор каждой главы.

Глава 1, “Введение в WPF”. В ней описывается архитектура WPF, основы DirectX и новая система измерений, не зависящая от устройств, которая автоматически изменяет размеры пользовательских интерфейсов.

Глава 2, “Язык XAML”. В этой главе рассматривается язык XAML, который вы будете применять для определения пользовательских интерфейсов. Вы узнаете о том, почему он был создан, и как он работает, а также создадите базовое окно WPF с помощью разных приемов написания кода.

*Глава 3, “Класс *Application*”.* В ней рассказывается о модели приложений в WPF. Вы увидите, как создаются отдельные экземпляры WPF-приложений на основе документов, и узнаете о том, как они используют потоковую модель WPF.

Глава 4, “Компоновка”. Эта глава посвящена панелям компоновки, которые позволяют организовать элементы в окне WPF. Мы рассмотрим разные стратегии компоновки и создадим некоторые обычные типы окон.

Глава 5, “Содержимое”. В ней описывается модель элементов управления содержимым, которая позволяет помещать элементы в *других* элементах, чтобы настроить внешний вид обычных элементов управления, таких как кнопки и метки.

Глава 6, “Свойства зависимостей и маршрутизируемые события”. Здесь будет рассказано о том, как WPF расширяет систему свойств и событий .NET. Вы увидите, как WPF использует свойства зависимостей для поддержки ключевых функциональных особенностей, таких как привязка данных и анимация, и как она применяет маршрутизацию событий для отправки передающихся вверх и туннельных событий через элементы вашего пользовательского интерфейса.

Глава 7, “Классические элементы управления”. В ней будут рассмотрены некоторые обычные элементы управления, с которыми знаком каждый разработчик Windows-приложений. К ним относятся кнопки, текстовые окна и метки, а также их характерные отличия в WPF.

Глава 8, “Окна”. В этой главе мы поговорим о том, как работают окна в WPF. Вы узнаете также, каким образом создаются окна с нерегулярными формами и как используются “стеклянные” эффекты, являющиеся визитной карточкой интерфейса Vista.

Глава 9, “Страницы и навигация”. Здесь будет описан процесс создания страниц в WPF и отслеживание хронологии посещения страниц. Вы увидите также, как можно создать WPF-приложение, размещаемое в браузере, которое можно запускать на Web-сайте, не выполняя утомительную процедуру инсталляции.

Глава 10, “Команды”. Эта глава посвящена модели команд WPF, позволяющей связывать множество элементов управления с одним и тем же логическим действием.

Глава 11, “Ресурсы”. В этой главе речь идет о том, как внедрять бинарные файлы в вашу сборку с помощью ресурсов, и как многократно использовать важные объекты в вашем пользовательском интерфейсе.

Глава 12, “Стили”. Эта глава посвящена системе стилей WPF, которая позволяет применять набор обычных значений свойств к целой группе элементов управления.

Глава 13, “Фигуры, трансформации и кисти”. Эта глава рассказывает о модели двухмерного рисования в WPF. Вы узнаете о том, как создаются формы, как изменяются элементы с помощью преобразований, и как рисуются экзотические эффекты с градиентами, мозаикой и изображениями.

Глава 14, “Классы Geometry, Drawing и Visual”. В этой главе более глубоко рассматривается двухмерная модель рисования. Вы узнаете о том, как создаются сложные пути, включающие дуги и кривые, как эффективно используется сложная графика, и как применяется низкоуровневый визуальный слой для оптимизированного рисования.

Глава 15, “Шаблоны элементов управления”. В ней будет показано, как можно придать любому элементу управления WPF совершенно новый внешний вид (и новое поведение), подключая специально созданный шаблон. Также вы увидите, как с помощью шаблонов можно создавать приложения со сменными обложками.

Глава 16, “Привязка данных”. Эта глава рассказывает о привязке данных в WPF. Вы увидите, как можно привязать любой тип объекта к пользовательскому интерфейсу, будь то экземпляр специального класса данных или полноценный DataSet из ADO.NET. Вы узнаете также о том, как выполняется преобразование, форматирование и проверка данных.

Глава 17, “Шаблоны данных, представления данных и поставщики данных”. В ней будет рассказано о хитростях, которые можно применять при проектировании профессиональных интерфейсов, управляемых данными. По ходу дела будут созданы расширенные списки данных, включающие рисунки, элементы управления и эффекты выделения.

Глава 18, “Списки, деревья, панели инструментов и меню”. Здесь рассматривается семейство элементов управления списками в WPF. Вы познакомитесь с элементами управления, ориентированными на данные, такими как сетки и деревья, а также с элементами управления, ориентированными на команды, такими как панели инструментов и меню.

Глава 19, “Документы”. В ней будет рассмотрена поддержка расширенных документов в WPF. Вы узнаете о том, как используются потоковые документы для представления больших объемов текста в наиболее удобной для чтения форме, и будете работать с фиксированными документами для отображения страниц, готовых к печати. А с помощью элемента управления RichTextBox можно предоставить пользователям возможность редактировать документы.

Глава 20, “Печать”. Посвящена модели печати в WPF, с помощью которой вы можете “рисовать” текст и формы в печатаемом документе. Вы узнаете также о том, как производится управление параметрами настройки страницы и очередью печати.

Глава 21, “Анимация”. В этой главе рассматривается структура анимации в WPF, которая позволяет интегрировать динамические эффекты в приложение с помощью простой декларативной разметки.

Глава 22, “Звук и видео”. Здесь будет описана поддержка медиа-информации в WPF. Вы увидите, как управлять воспроизведением звука и видео, и узнаете, как получать синхронизированную анимацию и создавать живые эффекты.

Глава 23, “Трехмерная графика”. В ней рассматривается поддержка рисования трехмерных форм в WPF. Вы узнаете о том, как создаются, преобразуются и ани-

мируются трехмерные объекты. Вы увидите даже, как помещать интерактивные двухмерные элементы управления на трехмерных поверхностях.

Глава 24, “Пользовательские элементы”. Здесь мы поговорим о том, как расширять существующие элементы управления в WPF, и как создавать собственные элементы управления. Вы увидите несколько примеров, включая средство для выбора цвета, основанное на шаблоне, маскированное текстовое поле, а также декоратор, с помощью которого выполняется специальное рисование.

Глава 25, “Взаимодействие с Windows Forms”. В этой главе вы узнаете о том, как комбинировать содержимое WPF и Windows Forms в одном и том же приложении, и даже в одном и том же окне.

Глава 26, “Развертывание ClickOnce”. Эта глава покажет, как развертывать WPF-приложения с помощью модели ClickOnce, появившейся в .NET 2.0.

Что необходимо для использования этой книги

Для того чтобы запускать WPF-приложения, ваш компьютер должен работать под управлением Microsoft Windows Vista или Microsoft Windows XP с пакетом обновлений Service Pack 2. Чтобы создавать WPF-приложения (и открывать примеры проектов, которые прилагаются к этой книге), вам необходимо использовать одну из следующих версий Visual Studio:

- Visual Studio 2005 с расширениями .NET 3.0 и WPF;
- следующую версию Visual Studio (под кодовым названием “Orcas”).

Чтобы точно знать, что вам понадобится для инсталляции расширений .NET 3.0 и WPF, зайдите на страницу этой книги по адресу <http://www.prosetech.com>.

Можно поступить и по-другому. Вместо того чтобы устанавливать какую-нибудь версию Visual Studio, вы можете использовать Expression Blend — графически ориентированный инструмент проектирования — для создания и тестирования WPF-приложений. В общем случае, Expression Blend предназначен для графических дизайнеров, занимающихся серьезными и красивыми по внешнему виду проектами, в то время как Visual Studio идеально подходит для программистов, создающих приложения на основе кода. Эта книга предполагает, что вы имеете дело с Visual Studio. Если вы желаете узнать больше о Expression Blend, можете прочитать одну из многих книг, посвященных этому продукту, включая *Pro Expression Blend*, написанную Дареном Лютцем (Darren Lutz) и Райаном Муром (Ryan Moore) (Apress, 2007 г.).

В некоторых примерах этой книги используется код доступа к данным ADO.NET для запроса к базе данных SQL Server. Чтобы иметь возможность работать с этими примерами, можете воспользоваться файлом сценария, включенным в загружаемый код, чтобы инсталлировать базу данных (в SQL Server 2000 или более поздней версии). Как вариант, вы можете использовать файловый компонент базы данных, который также входит в состав загружаемого кода. Этот компонент получает такие же данные из файла XML, имитируя работу полного компонента базы данных, не запрашивая при этом живой экземпляр SQL Server.

Visual Studio Orcas

На момент написания этой книги версия Visual Studio Orcas пребывала на этапе бета-тестирования. Если вы выбрали ее, вам придется преобразовать примеры проектов, прилагаемые к этой книге, прежде чем вы сможете открыть их. Когда вы откроете один из этих проектов в первый раз, Visual Studio запустит мастер преобразования, который поможет выполнить этот процесс. Процесс преобразования не изменяет сам код в этих примерах — он всего лишь обновляет файлы `.csproj` таким образом, чтобы можно было работать в новом формате.

Visual Studio Orcas обладает одним очевидным преимуществом по сравнению с Visual Studio 2005 — он поддерживает среду времени проектирования, которая позволит создавать пользовательские интерфейсы посредством операций, выполняемых с помощью мыши, как при разработке приложений с применением Windows Forms. С помощью Visual Studio 2005 вы можете предвзительно просматривать полученные пользовательские интерфейсы на этапе проектирования, однако вам придется писать слишком много кода вручную.

С технической точки зрения, Visual Studio 2005 использует .NET 3.0, которая включает WPF 1.0. Visual Studio Orcas включает .NET 3.5 со слегка обновленной версией WPF. Эта новая версия WPF, еще не вышедшая официально, имеет несколько существенных изменений. Ключевым отличием является исправление ошибок, настройка производительности и поддержка LINQ (набор языковых расширений в .NET 3.5, с помощью которых можно выполнять запросы, подобные SQL).

Примеры кода и URL-адреса

Рекомендуется посетить сайт издательства или зайти на страницу <http://www.prosetech.com> и загрузить примеры кода для этой книги. Это нужно для того, чтобы вы смогли самостоятельно проработать большинство самых сложных примеров кода, поскольку менее важные детали обычно остаются за кадром. Эта книга сфокусирована на наиболее важных разделах, поэтому вам не придется понапрасну перелистывать страницы, чтобы понять суть.

От издательства

Вы, читатель этой книги, и есть главный ее критик и комментатор. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересно услышать и любые другие замечания, которые вам хотелось бы высказать в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо, либо просто посетить наш Web-сервер и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится или нет вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию последующих книг. Наши координаты:

E-mail: info@williamspublishing.com

WWW: <http://www.williamspublishing.com>

Информация для писем из:

России: 115419, Москва, а/я 783

Украины: 03150, Киев, а/я 152