به نام خداوند گسترده مهر مهربان

.NET Compact Framework

برنامه نویسی ویندوز موبایل

.NET Compact Framework

مترجم: مهدی محبیان

منبع : MSDN

فهرست مطالب

.NET Compact Framework

.NET Compact Framework Microsoft یک کامپوننت یکپارچه روی دستگاه‌های Windows CE و Windows Mobile است که شما را قادر می‌سازد تا برنامه‌های مدیریت شده را ساخته و اجرا کنید و از وب سرویس‌ها استفاده نمایید. .NET Compact Framework دربردارنده یک CLR بهینه‌سازی شده و زیرمجموعه‌ای از کتابخانه کلاس .NET Framework است که ویژگی‌هایی مانند Windows Communication Foundation (WCF) و Windows Forms را پشتیبانی می‌کند. ضمناً این چارچوب حاوی کلاس‌هایی است که منحصراً برای .NET Compact Framework طراحی شده‌اند.

.NET Compact Framework معماری کامل .NET Framework از CLR و اجرای کد مدیریت شده را ارث می‌برد. برای آگاهی از اطلاعات بیشتر درباره معماری .NET Compact Framework، بخش «نگاه اجمالی بر معماری .NET Compact Framework» را ملاحظه نمایید.

.NET Compact Framework از توسعه برنامه‌ها در Visual Basic و Visual C# پشتیبانی می‌کند. این چارچوب فعلاً از توسعه برنامه‌ها در C++ پشتیبانی نمی‌کند.

نکته:

در این مستندات فرض بر این است که یک شناخت عمومی از .NET Framework وجود دارد. این مستندات روی اجزا و تکنولوژی‌هایی تمرکز می‌کند که دارای اهمیت خاصی بوده و یا منحصر به .NET Compact Framework هستند. مستندات .NET Compact Framework عناوینی را که به طور مشترک با .NET Framework داراست، تکرار نمی‌کند. برای مثال، .NET Compact Framework از همان مستندات کتابخانه کلاس استفاده می‌کند که .NET Framework کامل آنها را به کار می‌برد.

ویژوال استودیو و .NET Compact Framework

.NET Compact Framework Microsoft به طور کامل در میان .NET 2003 Visual Studio و نسخه‌های بعدی آن گنجانده شده است، به شرطی که گزینه نصب Smart Device Programmability برای توسعه برنامه‌های Visual

Basic و #Visual C انتخاب شده باشد. این گزینه به طور پیش فرض انتخاب شده می‌باشد. .NET Compact Framework فعلاً از توسعه برنامه‌های C++ پشتیبانی نمی‌کند.

توسعه و گسترش برنامه در ویژوال استودیو

Microsoft Visual Studio 2008 نوع پروژه Smart Device را به منظور توسعه برنامه‌ها برای Pocket PC، Smartphone و دیگر پلت‌فرمهای مبتنی بر Windows CE تدارک دیده است. چنان چه دستگاه هوشمندی (Smart Device) نداشته باشید، می‌توانید با استفاده از تکنولوژی نمونه‌سازی (Emulation)، بدون این که محیط توسعه یکپارچه Visual Studio 2008 را ترک کنید، برنامه‌های دستگاه هوشمند خود را ایجاد کرده و آزمایش نمایید.

شما می‌توانید از Visual Studio .NET 2003 Professional یا نسخه‌های بعدی آن برای طراحی برنامه‌های .NET Compact Framework برای دستگاه‌های مبتنی بر نسخه‌های پلت‌فرم زیر استفاده کنید:

- Windows Mobile 2003 for Pocket PC
- Windows Mobile 2003 for Smartphone
- Windows CE 5.0

در Microsoft Visual Studio 2008 می‌توانید با نصب SDKهای مربوطه، برنامه‌هایی را برای نرم‌افزار Windows Mobile نسخه 5.0 برای Pocket PCها و Smartphoneها توسعه دهید. برای آگاهی از دستورالعمل‌های نصب، بخش «منابع خارجی برای .NET Compact Framework» را ملاحظه نمایید. توجه نمایید که .NET Compact Framework نسخه 2.0 برای توسعه Smartphone نیازمند نرم‌افزار Windows Mobile 5.0 برای Smartphone است.

نگاه اجمالی بر معماری .NET Compact Framework

.NET Compact Framework برای کارکرد بهینه تحت قیود منابع دستگاه محدود طراحی شده است. این بخش مفاهیم کلیدی معماری .NET Compact Framework، مدیریت حافظه و ملاحظات امنیتی را شرح می‌دهد.

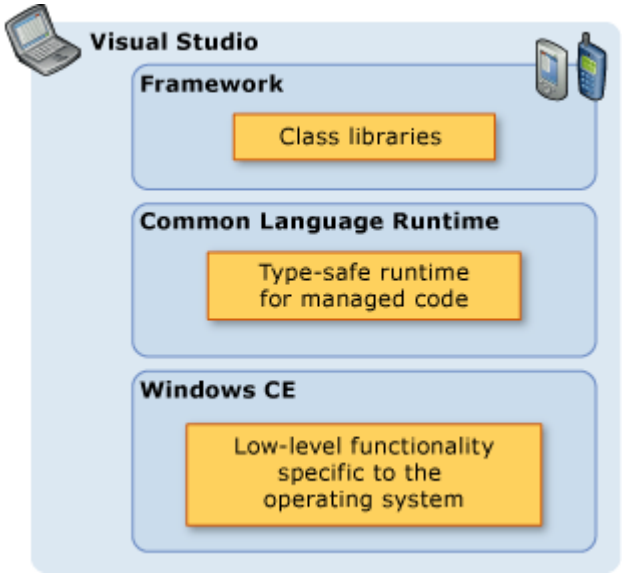
معماری .NET Compact Framework

.NET Compact Framework. معماری .NET Framework. کامل CLR را برای اجرای کد مدیریت شده، ارث می‌برد. .NET Compact Framework. قابلیت همکاری و تعامل با سیستم عامل Windows CE یک دستگاه را ارائه می‌دهد به طوری که شما می‌توانید به توابع خالص (native functions) دسترسی پیدا کرده و کاپوننت‌های (Components) خالص مورد علاقه خود را در میان برنامه‌تان بگنجانید.

شما می‌توانید برنامه‌های خالص و مدیریت شده را یک جا و با هم اجرا نمایید. میزبان قلمرو برنامه، که خود یک برنامه خالص است، نمونه‌ای از CLR را برای اجرای کد مدیریت شده راه‌اندازی می‌کند.

معماری

شکل زیر معماری پلت‌فرم .NET Compact Framework. را به طور مختصر بیان می‌کند.



Windows CE

.NET Compact Framework. از سیستم عامل Windows CE برای کارکرد هسته‌ای و برخی از ویژگی‌های خاص دستگاه استفاده می‌کند. برخی از نوع‌ها و اسمبلی‌ها، مانند آنهایی که برای Windows Forms، گرافیک‌ها، ترسیم و وب سرویس‌ها هستند، از نو ساخته شده‌اند تا به جای کپی شدن از .NET Framework. کامل، به شکلی کارآمد بر روی دستگاه‌ها اجرا شوند.

.NET Compact Framework. قابلیت همکاری و تعامل زیر را با Windows CE ارائه می‌دهد:

- سازگاری با امنیت خالص.
- به هم پیوستگی کامل با برنامه های نصب خالص.
- قابلیت همکاری و تعامل با کد خالص با استفاده از عملیات متقابل COM و احضار پلت‌فرم.

Common Language Runtime (CLR)

CLR مربوط به .NET Compact Framework. نیز از نو ساخته شده تا به منابع تحمیلی اجازه دهد تا روی حافظه محدود اجرا شده، و به شکلی کارآمد از برق باتری استفاده کند.

یک لایه انطباق پلت‌فرم، که در شکل قبل نشان داده نشده است، مابین Windows CE و CLR وجود دارد تا سرویس‌ها و واسط‌های دستگاه مورد نیاز توسط CLR و Framework را روی واسط‌ها و سرویس‌های Windows CE نگاشته کند.

Framework

.NET Compact Framework. زیر مجموعه‌ای از .NET Framework. بوده و گذشته از این دربردارنده ویژگی‌هایی است که به طور انحصاری برای .NET Compact Framework. طراحی شده‌اند. .NET Compact Framework. ویژگی‌ها و راحتی استفاده‌ای را عرضه می‌کند که دستیابی طراحان برنامه دستگاه خالص را به .NET. Framework و دستیابی طراحان برنامه رومیزی را به دستگاه‌ها آسان می‌کنند.

Visual Studio

تجربه توسعه برنامه‌های دستگاه هوشمند با Microsoft Visual Studio 2008 به سادگی برنامه‌های رومیزی است. توسعه دستگاه هوشمند در ویژوال استودیو دربردارنده مجموعه‌ای از انواع پروژه‌ها و نمونه‌سازهاست که توسعه برنامه برای Pocket PC، Smartphone و Windows CE تعبیه شده را هدف می‌گیرند.

حوزه‌های برنامه در .NET Compact Framework

هر برنامه .NET Compact Framework در میان یک ساختار زمان اجرا به نام یک حوزه (Domain) برنامه، که مشابه یک فرایند سیستم عامل است، اجرا می‌شود. .NET Compact Framework تضمین می‌کند که تمامی منابع مدیریت شده استفاده شده توسط یک برنامه در حال اجرا، زمانی که برنامه خاتمه یابد آزاد شده یا به سیستم عامل میزبان برگردانده می‌شوند.

حوزه‌های برنامه بسیاری از مزایای فرایندها مانند جداسازی خطا و اشتباه، استحکام بهبود یافته و امنیت را، بدون نیاز به پشتیبانی از سیستم عامل میزبان متضمن تقدیم می‌کند. یک میزبان حوزه برنامه، نمونه‌ای از CLR را راه‌اندازی می‌کند و خودش کد سیستم عامل خالص است. CLR می‌تواند به طور استاتیک یا پویا به میزبان حوزه برنامه پیوند داده شود.

.NET Compact Framework هیچ محدودیتی را روی رفتار میزبان حوزه برنامه قرار نمی‌دهد. میزبان حوزه برنامه می‌تواند یک تعمیم ساده به پوسته تعاملی موجود باشد که برای راه‌اندازی و توقف برنامه‌ها به کار برده می‌شود. روی سیستم‌های برنامه دینامیک مانند ویندوز، میزبان حوزه برنامه می‌تواند یک تعمیم به بارکننده برنامه باشد به طوری که برنامه‌های .NET Compact Framework می‌توانند با استفاده از مکانیزم مشابه با یک برنامه خالص به کار افتاده و متوقف شوند.

حوزه‌های برنامه‌ای متعدد

.NET Compact Framework از حوزه‌های برنامه‌ای متعدد پشتیبانی می‌کند. شما می‌توانید یک اسمبلی را در سازنده (Constructor) یک کلاس تعیین کنید. در این صورت می‌توانید متد CreateDomain را برای شروع یک حوزه برنامه جدید به کار ببرید. حوزه برنامه جدید کپی‌های مال خود را از DLLهای زمان اجرای زبان مشترک (CLR)، ساختارهای داده‌ای و منابع حافظه بارگیری می‌کند. حوزه‌های برنامه‌ای متعدد می‌توانند در یک فرایند سیستم عامل وجود داشته باشند.

نکته:

.NET Compact Framework از بارگذاری اسمبلی‌ها در میان ناحیه کد بی اثر حوزه برای استفاده توسط

حوزه‌های برنامه‌ای متعدد، پشتیبانی نمی‌کند.

.NET Compact Framework مشخص می‌کند که Garbage Collection چه زمانی لازم است تا اجرا شود. Garbage Collection می‌تواند در یک حوزه برنامه یکتا یا در تمامی حوزه‌های برنامه رخ دهد. این امر مانع می‌شود تا یک حوزه برنامه حافظه خیلی زیادی را به خرج دیگران مصرف کند.

مدیریت حافظه دستگاه در .NET Compact Framework

یک توانایی مهم .NET Compact Framework کاربرد منابع به صورت کارآمد است به خصوص استفاده از RAM فرار. دستگاه‌ها لزومی ندارد که دارای واحدهای مدیریت حافظه (MMU) سخت‌افزاری یا حافظه مجازی سیستم عامل باشند.

اندازه انباره ذخیره‌سازی .NET Compact Framework

برای .NET Compact Framework نسخه 2.0، اندازه انباره ذخیره‌سازی به صورت زیر است:

- 5.5 MB (ROM)، به صورت غیر فشرده، روی Windows Mobile 5.0.

نکته:

انباره ماندگار حین راه‌اندازی با Windows Mobile 5.0 دیگر غیر فشرده نخواهد بود.

برای .NET Compact Framework نسخه 1.0، اندازه انباره ذخیره‌سازی به صورت زیر است:

- 1.55MB (ROM)، به صورت فشرده، روی Pocket PC 2000/2002
- 1.35MB (ROM)، به صورت فشرده، روی Windows Mobile 2003 برای Pocket PC و Windows Mobile 2003 برای Smartphone.

نیازمندی‌های RAM جاری و درحال کار:

- 5 MB+. (بستگی به برنامه دارد)

اندازه‌های نوعی برنامه:

- 5 – 100 KB

نصب فایل CAB

در طی نصب .NET Compact Framework نسخه 2.0 SP1، با پرامپتی از شما درخواست می‌شود تا آن را یا روی دستگاه یا روی یک کارت ذخیره‌سازی، در صورت موجود بودن نصب نمایید. چنان چه خواستید تا روی کارت ذخیره‌سازی نصب کنید، در این صورت نهانگاه اسمبلی سراسری (Global Assembly Cache) روی یک کارت ذخیره‌سازی جای داده می‌شود و فایل‌های مربوط به CLR در دایرکتوری Windows\ قرار داده می‌شوند. نهانگاه اسمبلی سراسری به طور تقریبی نیازمند 4.5 MB فضا بوده و CLR نیاز به 1.0 MB دارد.

بسته به پیاده‌سازی حافظه در سخت‌افزار و نرم‌افزار، عملکرد .NET Compact Framework. برای نصب فایل CAB متغیر است. برای برخی از دستگاه‌ها، بهترین عملکرد می‌تواند با نصب نهانگاه اسمبلی سراسری در یک کارت ذخیره‌سازی، به دست آید.

نحوه استعمال حافظه

.NET Compact Framework. طراحی شده است تا به صورت بهینه‌ای با مشخصات زیر، روی سیستم‌ها عمل کند:

- باتری دارای نیرو
- به طور تقریبی ۵ تا ۱۰ نوبت ظرفیت سیستم فایلی RAM یا حافظه فلش برای ذخیره .NET Compact Framework و برنامه‌های آن در RAM دینامیک (DRAM)
- یک فضای کاری از 128 KB تا 1 MB در DRAM
- درایو سخت (Hard Drive) اختیاری

.NET Compact Framework. از حافظه سیستم در دسترس با دقت استفاده می‌کند. تا زمانی که یک برنامه را اجرا کنید، RAM مورد دسترسی واقع نمی‌شود. علاوه بر این، .NET Compact Framework. زمانی که برنامه‌ها را

ترک می‌کنید، RAM را آزاد می‌کند. سیستم عامل خالص لزومی ندارد که امکانات حفاظت از حافظه مال خود را داشته باشد. استثناها همواره زمانی پرتاب می‌شوند که حافظه غیر تملیکی مورد دسترسی واقع شود.

زمانی که حافظه کم است، NET Compact Framework. به طور مداخله گرانه‌ای ساختارهای داده‌ای درونی را که توسط کدی که درحال اجرای فعلی مورد نیاز نیستند، آزاد می‌کند. بنابراین، برنامه می‌تواند به اجرایش ادامه دهد حتی در وضعیت‌های حافظه پایین. اگر برنامه نیازمند حافظه بیشتری از آن چه در دسترس است باشد، NET Compact Framework. برنامه را به طور تروتمیزی بسته و تمامی منابع متضمن را آزادسازی می‌کند. NET. Compact Framework خودش نبایستی به دلیل حافظه اندک عقیم بماند.

میزبان حوزه برنامه برنامه‌های NET Compact Framework. و CLR را راه‌اندازی می‌کند. برنامه‌های NET. Compact Framework از فضای کد و فضای داده دینامیک و استاتیک در وضعیت یکسان با برنامه‌های خالص، استفاده می‌کنند. زمانی که هیچ برنامه NET Compact Framework. در حال اجرا نباشد، در آن جا هیچ گونه هزینه RAM خارج از میزبان حوزه برنامه وجود نخواهد داشت، و اندازه کمی از داده استاتیک توسط CLR به کار برده خواهد شد. زمانی که یک برنامه NET Compact Framework. راه‌اندازی می‌شود، NET. Windows CE یک میزبان حوزه برنامه ایجاد می‌کند.

The .NET Compact Framework applications are packaged into .exe files and .dll files, which can be stored in a read-only or read/write file system in flash (or ROM for read-only). The common language runtime class loader can read these files in directly addressable blocks without making a memory copy and without requiring a memory management unit to create a memory-mapped view of the file.

برنامه‌های NET Compact Framework. در میان فایل‌های exe. و فایل‌های dll. بسته‌بندی می‌شوند که این فایل‌ها می‌توانند در یک سیستم فایل فقط–خواندنی یا خواندنی/نوشتنی در فلش (یا ROM برای فقط–خواندنی) ذخیره شوند. بارکننده کلاس CLR می‌تواند این فایل‌ها را به طور مستقیم در بلوک‌های آدرس‌پذیر، بدون انجام یک کپی از حافظه و بدون نیاز به واحد مدیریت حافظه (MMU) به منظور ایجاد یک نمای نگاشت حافظه از فایل، بخواند.

توسعه دهندگان برنامه ترغیب می‌شوند تا برنامه‌ها را روی دستگاه‌های متعددی آزمایش کنند تا اختلافات مختص دستگاه‌ها را از نظر کارکرد درک کنند.

انباره RAM

حافظه با دسترسی تصادفی (Random access memory (RAM)) برای ذخیره‌سازی ساختارهای داده دینامیک و کد کامپایل شده JIT استفاده می‌شود. .NET Compact Framework از RAM در دسترس تا حد مشخص شده توسط دستگاه برای پنهان کردن کد تولید شده و ساختارهای داده استفاده می‌کند و پس از آن هر زمان که مناسب باشد، حافظه را آزاد می‌کند.

The common language runtime uses a code-pitching technique to free blocks of JIT-compiled code at run time when memory is low. This enables larger programs to run on RAM-constrained systems with minimal performance penalty.

زمانی که حافظه پایین باشد، CLR از یک تکنیک code-pitching برای آزادسازی بلوک‌های کد کامپایل شده JIT در زمان اجرا استفاده می‌کند. این امر برنامه‌های بزرگ‌تر را قادر می‌سازد تا روی سیستم‌های مقید به RAM، با کمترین جریمه کارکردی اجرا شوند.

انباره ROM

کد خالصی که CLR را شکل می‌دهد، می‌تواند در حافظه فقط خواندنی (ROM) یا یک سیستم فایل RAM مقیم شود. .NET Compact Framework از ROM در دسترس، فلش یا مخزن دیسک استفاده می‌کند تا برنامه‌ها را قادر سازد تا در وضعیت‌های حافظه پایین، با کمترین کارایی به اجرا شدن ادامه دهند.

فایل‌هایی که دربردارنده دستورالعمل‌های زبان طبقه میانی مایکروسافت (Microsoft Intermediate Language (MSIL)) و داده‌های جانبی (Metadata) برای کتابخانه‌های کلاس هستند، یا در یک سیستم فایل ROM یا در یک سیستم فایل RAM ذخیره می‌شوند. کتابخانه‌های کلاس می‌توانند به صورت بخشی از فرایند نصب برنامه قابل دانلود، در میان یک سیستم فایل خواندنی/نوشتنی بارگذاری شوند.

امنیت در .NET Compact Framework

در .NET Compact Framework، سیاست‌گذاری امنیتی عهده‌دار یک پلت‌فرم باز شده و اعتماد کاملی به تمامی کدها می‌بخشد. نسخه‌های بعدی .NET Compact Framework ممکن است زیرمجموعه‌ای از سیاست گزینشی و امنیت دسترسی به کد مبتنی بر شواهد .NET Framework کامل را عرضه کنند. سیاست امنیتی در نسخه‌های

بعدی .NET Compact Framework. ممکن است بسته به دستگاه و نیز حوزه امنیتی (Security Domain) تغییر کند.

کدی که از احضار پلت‌فرم استفاده کرده و به سیستم فایل دسترسی پیدا می‌کند روی Pocket PC باز کار خواهد کرد، اما ممکن است روی یک دستگاه OEM Windows CE با امنیت مقفول از کار افتاده، یک استثنای امنیتی پرتاب کند.

ملاحظات امنیتی مربوط به دستگاهها

چندین ملاحظات امنیتی منحصر به فرد و مهم برای .NET Compact Framework. وجود دارد:

- .NET Compact Framework. فعلاً از امنیت دسترسی به کد پشتیبانی نمی‌کند. هنگام تبدیل برنامه‌ها از .NET Framework. رومیزی کامل، کد زیر بایستی حذف شود:
  - اشیای مجوز امنیتی (Security permission objects).
  - درخواست‌های امنیتی.
  - هرگونه شواهد، از جمله Strong name ها.
  - گروه‌های کدی.
- نوعهای .NET Compact Framework. از مشخصه AllowPartiallyTrustedCallersAttribute پشتیبانی نمی‌کنند. در کتابخانه کلاس .NET Framework. کامل، بسیاری از عناوین مرجع عضوی دربردارنده هشدارهایی هستند که عضو نیازمند اعتماد کامل برای فراخواننده فوری است. از آن جایی که .NET Compact Framework از AllowPartiallyTrustedCallersAttribute پشتیبانی نمی‌کند، به طور پیش فرض دسترسی نامحدودی به تمامی کتابخانه‌ها اعطا می‌شود. نسخه‌های بعدی .NET Compact Framework حاوی درخواست‌هایی برای اجازه‌های ویژه به دسترسی محدود به مشخصه‌های با اعتماد بالا خواهند شد. در این صورت پلیس امنیتی، هر جا که مناسب باشد، مجوزهای ویژه‌ای به کد اعطا خواهد کرد.

- دستگاهها می‌توانند مهندسی شوند تا به روزرسانی‌های مربوط به نرم‌افزار سیستم را پذیرفته و تحت کنترل تدابیر امنیتی خارج از دامنه .NET Compact Framework باشند.

- چنان چه دستگاه کد خالصی را پشتیبانی کند که غیر قابل اعتماد باشد، امنیت کامل نمی‌تواند قابل دسترسی باشد. تنفیذ امنیتی تنها از نظر قابلیت تصدیق امنیت، برای کد با نوع ایمن معتبر است. مرز مابین کد مدیریت شده و مدیریت نشده، وجود یک تهدید امنیتی جدی برای کد مدیریت شده است.

- در .NET Compact Framework نسخه 1.0، زمان اجرا (Runtime) احضار پلت‌فرم را از دسترسی به کد خالص باز نمی‌دارد.

چندین سازنده دستگاه مفاهیم پایگاه متقاطع (Cross-platform) را عرضه می‌کنند:

- سازنده تجهیزات اصلی (OEM) تراز امنیتی پلت‌فرم دستگاه و سیاست امنیتی آن را تعیین کرده و هم چنین می‌تواند یک مسئول امنیتی را همراه با مجوزهای سفارشی به منظور اعطای دسترسی، تدارک ببیند.

- گواهی Strong-name می‌تواند در میان پلت‌فرمها ناسازگار بوده و برنامه‌ها را تحت فشار قرار دهد. گرچه، یک سیاست انقیاد امنیتی روی هر پلت‌فرم می‌تواند نیازمندی اغلب برنامه‌ها برای تقاضای گواهی Strong-name اسمبلی‌های استاندارد را کم کند.

- فروشندگان نرم‌افزار مستقل می‌توانند مجموعه‌ای از تقاضاهای امنیتی را از میان کد متعلق به آنها بدون نیاز به اعمال تغییرات به سیاست امنیتی روی دستگاه، تحمیل کنند.

تدابیر امنیتی بر طبق نوع دستگاه و چگونگی به کار رفتن آن، تغییر می‌کنند. .NET Compact Framework قابلیت انتقال دستگاه متقاطع (Cross-Device) را اصلاح می‌کند، در حالی که شیوه‌های امنیتی قوی را نگه‌داری می‌کند.

اهداف امنیتی برای .NET Compact Framework

طرحهای امنیتی آینده برای .NET Compact Framework می‌تواند حاوی موارد زیر باشد:

- امنیت دسترسی به کد.
- یک سیاست امنیتی محدود به کد مدیریت شده روی دستگاه. گرچه، چنان چه پلت‌فرم به پشتیبانی از کد خالص ادامه دهد، امنیت پلت‌فرم کامل نمی‌تواند حاصل شود.
- گواهی اسمبلی سفارشی و سیاست امنیتی با زمینه قابل تعویض.
- انواع مجوز سفارشی برای کمک به تعمیمات موبایل با برنامه‌ی ایمن .NET Compact Framework..
- انواع مجوز سفارشی معرفی شده توسط سازندگان دستگاه به منظور کمک به کارکرد ویژه پلت‌فرم ایمن. این مجوزهای سفارشی می‌توانند در فضاهای نامی که با .NET Compact Framework. کار می‌کنند، فراهم شوند.

نگاه اجمالی بر .NET Compact Framework

.NET Compact Framework. محیطی را تشکیل می‌دهد که در آن، برنامه‌های مدیریت شده روی دستگاه‌ها اجرا می‌شوند. .NET Compact Framework. دسترسی به ویژگی‌های متضمن دستگاه را عرضه می‌کند. علاوه بر این، برنامه‌ها و اجزا (Components) می‌توانند روی دستگاه و در سرتاسر اینترنت فعل و انفعال داخلی داشته باشند.

تفاوتهای مابین .NET Compact Framework. و .NET Framework.

.NET Compact Framework. زیرمجموعه‌ای از .NET Framework. کامل می‌باشد. .NET Compact Framework. تقریباً ۳۰ درصد از کتابخانه کلاس .NET Compact Framework. را پیاده‌سازی می‌کند و ضمناً حاوی ویژگی‌ها و کلاس‌هایی است که مختص موبایل و توسعه تعبیه شده هستند.

این بخش تمامی تفاوت‌های مابین دو چارچوب را تشریح نمی‌کند، اما ملاحظات پراهمیت مربوط به توسعه برنامه‌ها را لیست می‌کند.

حوزه‌های برنامه

.NET Compact Framework. فعلاً از بارگذاری اسمبلی‌ها در میان یک ناحیه کد با حوزه خنثی به منظور استفاده توسط حوزه های برنامه‌ای متعدد، پشتیبانی نمی‌کند. برای آگاهی از اطلاعات بیشتر، بخش «حوزه‌های برنامه در .NET Compact Framework.» را ملاحظه نمایید.

آرایه‌ها

اگر چه برخی از زبان‌ها از حدود پایین‌تر دیگری غیر از صفر پشتیبانی می‌کنند، CLR از این موضوع پشتیبانی نمی‌کند و چنان چه اولین عنصر صفر نباشد، یک استثنای MissingMethodException پرتاب می‌کند.

ASP.NET

.NET Compact Framework اصولاً یک پلت‌فرم پرمشتری بوده و پشتیانی ASP.NET را در اختیار قرار نمی‌دهد. به منظور توسعه صفحات وب برای دستگاههای موبایل، شما می‌توانید از کنترل‌های ASP.NET mobile Web استفاده کنید. به منظور توسعه صفحات وب برای کامپیوترهای شخصی یا تهیه کنندگان سرویس وب، مستندات ASP.NET خود را ملاحظه نمایید.

فرمت فایل‌ها و اسمبلی‌ها

برنامه‌های مربوط به هر دو چارچوب، از اسمبلی‌ها استفاده می‌کنند. هر دو چارچوب به فایل‌های قابل اجرای قابل حمل ((Portable Executable (PE دسترسی دارند، که این فایل‌ها حاوی زبان طبقه میانی مایکروسافت (MSIL) و داده‌های جانبی هستند که یک برنامه .NET Framework را تعریف می‌کند. یک فایل PE می‌تواند به یک فضای نام برنامه‌نویسی اشاره کند که توسط فایل‌های اسمبلی دیگر تعریف شده و به اشتراک گذاشته می‌شود. برای آگاهی از اطلاعات بیشتر، بخش «سازگاری‌های دودوئی با .NET Framework کامل» را ملاحظه نمایید.

اسمبلی‌ها و نهانگاه اسمبلی سراسری

.NET Compact Framework فعلاً از اسمبلی‌های چند پیمانه‌ای پشتیبانی نمی‌کند، اما از اسمبلی‌های اقماری پشتیبانی می‌کند.

کلاس‌ها و نوعها

.NET Compact Framework زیرمجموعه‌ای از کتابخانه کلاس .NET Framework را پشتیبانی می‌کند. این زیرمجموعه در خور برنامه‌هایی است که طراحی می‌شوند تا روی دستگاه‌های با منابع اجباری اجرا شوند و از نظر مفهومی و معناشناختی این زیر مجموعه سازگار با کلاس‌های با نام مشابه در .NET Framework است. برای

آگاهی از اطلاعاتی درباره چگونگی مشخص کردن پشتیبانی لازم برای .NET Compact Framework، بخش «چگونگی مشخص کردن اعضای پشتیبانی شده در کتابخانه کلاس» را ملاحظه نمایید.

عملیات متقابل COM

.NET Compact Framework نسخه 2.0 از عملیات متقابل COM پشتیبانی می‌کند. ضمناً دربردارنده امکانات مرتب‌سازی پیشرفته است. برای آگاهی از اطلاعات بیشتر، بخش «قابلیت همکاری و تعامل در .NET Compact Framework» را ملاحظه نمایید.

CLR

CLRها در هر دو چارچوب از تعمیم کد مدیریت شده، کامپایل کد JIT و تکنیک Garbage Collection بهره می‌برند. آنها از مشخصات زبان مشترک (Common Language Specification (CLS)) پشتیبانی می‌کنند. هر دو چارچوب نوعهای اصلی توکار و نیز نوعهای دیگری دارند که می‌توانید هنگام ساخت برنامه خود، آنها را به کار برده و از آنها استنتاج کنید. CLR مربوط به .NET Compact Framework به طور تقریبی برابر ۱۲ درصد اندازه CLR برای .NET Framework است.

کنترل‌ها

.NET Compact Framework اغلب کنترل‌های Windows Forms عرضه شده توسط .NET Framework کامل را پشتیبانی می‌کند و حاوی کنترل‌هایی است که خاص .NET Compact Framework هستند. کنترل‌های Windows Forms مخصوصاً برای .NET Compact Framework ساخته شده‌اند.

دایرکتوری جاری

قابلیت دایرکتوری جاری در سیستم عامل Windows Embedded CE وجود ندارد. بنابراین، .NET Compact Framework از متدهای GetCurrentDirectory و SetCurrentDirectory پشتیبانی نمی‌کند.

.NET Compact Framework از خاصیت WorkingDirectory برای یک شیء ProcessStartInfo، پشتیبانی می‌کند. گرچه، متن آن توسط برنامه اجرایی در حال اجرا، در بارگذاری‌های فایل و راه‌اندازی‌های بعدی حفظ نمی‌شود.

داده‌ها

.NET Compact Framework یک پیاده‌سازی زیرمجموعه‌ای از ADO.NET عرضه می‌کند و حاوی مهیا کننده داده SQL Server Mobile است. فضای نام System.Data.OleDb پشتیبانی نمی‌شود.

انواع داده و دقت ممیز شناور

.NET Compact Framework از برشمارش (ریزفهرست) MidpointRounding پشتیبانی نمی‌کند.

در یک محاسبه تقسیم، چنان چه مقسوم علیه خیلی بزرگ باشد یا در مقدار حداکثری ممیز شناور باشد، یا خیلی کم باشد یا در مقدار حداقلی ممیز شناور باشد، محاسبه به جای برآورد صحیح، نتیجه صفر را برمی‌گرداند.

پلت‌فرم MIPS از دقت کامل مشخص شده توسط *استاندارد مربوط به محاسبات ممیز شناور دودویی* (IEEE 754) پشتیبانی نمی‌کند، و می‌تواند باعث بروز نتایج غیر قابل پیش‌بینی شود. به علت ملاحظات مربوط به عملکرد، .NET Compact Framework قابلیت نمونه‌سازی ممیز شناوری را برای این پلت‌فرم در اختیار قرار نمی‌دهد.

اشکال‌زدایی در خط فرمان

.NET Compact Framework از دیباگر خط فرمان .NET Framework (یعنی MDbg.exe) که توسط .NET Framework 2.0 عرضه شده است، پشتیبانی نمی‌کند. دیباگر CLR قدیمی‌تر (یعنی DbgCLR.exe) در نسخه 2.0 هر دو چارچوب تقبیح شده است.

نماینده ها (Delegates)

نماینده‌های ناهماهنگ (وغیرهمزمان)، به خصوص متدهای **BeginInvoke** و **EndInvoke**، پشتیبانی نمی‌شوند.

گسترش دادن برنامه‌ها

برای گسترش یک برنامه، می‌توانید با استفاده از یک کابل از کامپیوتر رومیزی، پورت اینفرارد یا یک اینترنت بی‌سیم یا اتصال اینترانت، به راحتی اسمبلی را به دستگاه هدف کپی کنید. در ویژوال استودیو ۲۰۰۸، در حالی که اشکال‌زدایی می‌کنید، می‌توانید به طور مستقیم، (برنامه را) به دستگاه گسترش دهید.

ردیابی امکانات عیب شناسی

.NET Compact Framework از فایل‌های پیکربندی برای ردیابی پشتیبانی نمی‌کند، اما شما می‌توانید از شمارنده‌های عملکرد استفاده کنید.

اشیاء بی حفاظ

.NET Framework کامل تضمین نمی‌کند که دسترسی به خاصیت‌ها یا متدها روی یک شیء بی‌حفاظ همواره موفقیت آمیز خواهد بود. گرچه، دسترسی به برخی از خاصیت‌ها مانند **Text** اغلب اوقات روی .NET Compact Framework نتیجه‌بخش است. به علت اختلافات مابین دو چارچوب، دسترسی به خاصیت‌ها یا متدها روی یک شیء بی‌حفاظ روی .NET Compact Framework تقریباً همیشه نافرجام خواهد ماند.

رمزگذاری و محلی سازی

پشتیبانی از محلی‌سازی (جهانی‌سازی)، مانند فرمت داده‌ها و طبقه‌بندی جداول درخور برای منطقه، هر زمان که برای بازده اندازه و سازگازی امکان‌پذیر باشد، به سیستم عامل متضمن واگذار می‌شود.

.NET Compact Framework برای مرتب‌سازی داده‌ها وابسته به سیستم عامل است. از این رو، مرتب‌سازی (Sorting) می‌تواند برای برخی از فرهنگ‌ها، نتایج غیرمنتظره‌ای را تولید کند.

.NET Compact Framework از تنظیمات CurrentUICulture به ازای هر ریسمان (Thread) پشتیبانی نمی‌کند.

رویدادها

.NET Compact Framework از رویدادهای GotFocus و LostFocus پشتیبانی می‌کند، اما از رویدادهای Activated و Deactivated پشتیبانی نمی‌کند.

رشته‌های توصیف استثناء

NET Compact Framework. رشته‌های پیغام خطای استثنایی را در یک DLL مجزا به نام System.SR.dll به منظور صرفه‌جویی در حافظه، در اختیار قرار می‌دهد. هم چنین می‌توانید رشته‌های استثناء را برای دیگر فرهنگ‌ها، به منظور محلی ساختن برنامه‌ها تدارک ببینید.

مسیرها و اسامی فایل

Windows Embedded CE اسم فایلی را که بدون اطلاعات مسیر مشخص شده است، به صورتی تجزیه می‌کند که در دایرکتوری ریشه دستگاه باشد، نه در دایرکتوری برنامه. برای حصول اطمینان از موفقیت‌آمیز بودن عملیات‌ها، اطلاعات مسیر را به طور مطلق مشخص کنید.

NET Compact Framework. رشته‌های Uniform Resource Identifier (URI) را که با file:// پیشونددار شده‌اند، به شکل متفاوتی از NET Framework. پردازش می‌کند. یک تصریح نسبی مانند file://myfile به صورت \\myfile تجزیه و تحلیل می‌شود. رشته URI با سه فوروارد اسلش file:///myfile به صورت \myfile واقع در دایرکتوری ریشه، تجزیه می‌شود.

شما می‌توانید نسحه یک اسمبلی را با خاصیت **Version** به دست آورید، اما پشتیبانی از آن بستگی به سازنده دستگاه دارد و استفاده از آن نمی‌تواند تضمین شود. برای اخذ نام دایرکتوری که حاوی یک برنامه است، بخش «چگونگی تحصیل دایرکتوری برنامه» را ملاحظه نمایید.

تفکیک پذیری بالا

در NET Compact Framework. نسخه 2.0 و نسخه‌های بالاتر، رزلوشن (تفکیک‌پذیری) DPI به طور خودکار در پروژه‌های ویژوال استودیو مدیریت می‌شود. برای آگاهی از اطلاعات بیشتر، بخش «چگونگی اداره تغییرات جهت‌گیری و تفکیک‌پذیری» را ملاحظه نمایید.

ورودی/خروجی (I/O)

به علت تفاوت‌ها در سیستم‌های عامل دستگاه، محدودیت‌ها و قیودی در مورد مدل I/O وجود دارد. .NET Compact Framework تذکرات تغییر فایلی را در اختیار قرار نمی‌دهد. از آن جایی که I/O دستگاه در RAM اتفاق می‌افتد، مشخصه‌های فایل و دایرکتوری نمی‌توانند تنظیم شده یا مورد دسترسی واقع شوند.

نصب و فایل‌های CAB

شما می‌توانید از فایل‌های CAB استفاده کرده و برنامه‌های Microsoft Windows Installer را به منظور توزیع برنامه‌های خود، ایجاد کنید.

زبان‌ها

.NET Compact Framework از توسعه برنامه‌ها با استفاده از Visual Basic و Visual C# پشتیبانی می‌کند، اما فعلاً از C++ پشتیبانی نمی‌کند.

عملیات ریاضی

تمامی متدهای ریاضیاتی روی تمامی پلت‌فرم‌های دستگاه پشتیبانی نمی‌شوند؛ گرچه، آنها در API به منظور سازگاری ضمیمه شده‌اند.

حافظه

.NET Compact Framework برای سیستم‌هایی که از نیروی باتری استفاده می‌کنند بهینه‌سازی شده است و از استفاده سنگین از RAM و چرخش‌های CPU اجتناب می‌کند. برای آگاهی از اطلاعات بیشتر درباره چگونگی صرفه‌جویی در حافظه، بخش‌های «مدیریت حافظه دستگاه در .NET Compact Framework.» و «چگونگی بهبود عملکرد» را ملاحظه نمایید.

شبکه سازی

.NET Compact Framework کلاس‌های Infrared Data Association (IrDA) را برای ایجاد ارتباطات اینفرارد و کلاس‌های Web listening را به منظور سرویس‌رسانی درخواست‌های HTTP به دستگاه، در اختیار قرار می‌دهد.

این کلاس‌ها تنها در .NET Compact Framework در دسترس هستند. بخش «شبکه‌سازی و ارتباط در .NET Compact Framework» را ملاحظه نمایید.

بررسی عملکرد

.NET Compact Framework از پروفایل کردن کد یا فایل Perfmon.exe واقع در System Monitor پشتیبانی نمی‌کند. گرچه، شمارنده‌های عملکردی وجود دارند که می‌توانید از آنها استفاده کنید.

کد پراکسی

.NET Compact Framework از تمامی کدهای تولید شده توسط Web Services Description Language Tool (Wsdl.exe) پشتیبانی نمی‌کنند. به منظور تشخیص این که کدام کد پشتیبانی نمی‌شود، بخش «چگونگی استفاده از یک پراکسی تولید شده توسط Wsdl.exe» را ملاحظه نمایید.

بازتاب

.NET Compact Framework از فضای نام System.Reflection.Emit پشتیبانی نمی‌کند.

.NET Compact Framework فعلاً هنگام مقایسه اشیای بازتابی مانند MethodInfo، FieldInfo، PropertyInfo، EventInfo، MemberInfo، MethodBase، ConstructorInfo و ParameterInfo از عملگر برابری (==) پشتیبانی نمی‌کند. به عنوان یک جایگزین، «پیاده‌سازی متد Equals» را ملاحظه کنید.

هدایت از راه دور

.NET Compact Framework از قابلیت هدایت از راه دور (Remoting) پشتیبانی نمی‌کند. برای آگاهی از اطلاعات بیشتر درباره راه حل‌های جایگزین، بخش «صف‌بندی پیغام در .NET Compact Framework.» را ملاحظه نمایید.

مدیریت پیغام ایمن

.NET Compact Framework از گواهینامه‌های سمت مشتری (Client-side) و تعیین اعتبار با استفاده از HTTPS پشتیبانی نمی‌کند. به جای آن از تعیین اعتبار پایه‌ای (Basic Authentication) استفاده کنید.

امنیت

تفاوت‌های امنیتی و ملاحظات مربوط به آن، در بخش «امنیت در .NET Compact Framework» تشریح شده است.

سریالی‌سازی

به علت ملاحظات عملکردی و اندازه، .NET Compact Framework از سریالی‌سازی دودویی با استفاده از BinaryFormatter، یا سریالی‌سازی SOAP با استفاده از SoapFormatter پشتیبانی نمی‌کند.

با این حال، .NET Compact Framework پشتیبانی از سریالی‌سازی را برای ارسال داده شیء با استفاده از SOAP در سرویس‌های وب XML، و سریالی کردن مجموعه داده‌ها برای XML در اختیار قرار می‌دهد.

The .NET Compact Framework is 8 percent the size of the full .NET Framework redistributable package. The size on disk is 50 percent smaller because of Windows Embedded CE file system compression.

اندازه

.NET Compact Framework برابر با ۸ درصد اندازه پکیج قابل توزیع مجدد .NET Framework کامل است. این اندازه روی دیسک، به علت فشردگی سیستم فایل Windows Embedded CE، ۵۰ درصد کوچک‌تر است.

سوکت‌ها

تمامی گزینه‌های سوکت در .NET Compact Framework پشتیبانی نمی‌شوند. برای آگاهی از اطلاعات بیشتر، بخش «برنامه‌نویسی سوکت» را ملاحظه نمایید.

دست‌کاری رشته‌ها، عبارات منظم

برنامه‌هایی که از عبارات منظم در .NET Compact Framework استفاده می‌کنند، از نظر باینری سازگار با برنامه‌هایی نیستند که از عبارات منظم در .NET Framework کامل استفاده می‌کنند، اما از نظر کد منبع سازگارند.

ریسمان‌ها (Treads)

یک برنامه .NET Compact Framework حدود چهار ریسمان ایجاد می‌کند:

- یک ریسمان برنامه اصلی.

- یک ریسمان که برای کنترل تایمرهای دوره‌ای گوناگون و وقفه‌هایی که می‌توانند توسط سیستم یا برنامه‌ها زمان‌بندی شوند، به کار برده می‌شود.
- یک ریسمان که برای ردیابی تغییرات اعمالی به واسط‌های TCP/IP به کار برده می‌شود (با شبیه‌سازی رفتار رسانه گذاری که روی Windows XP موجود است اما روی Windows Embedded CE حضور ندارد).
- یک ریسمان که برای اجرای finalizerها به کار برده می‌شود. این ریسمان زمانی که اولین شیء قابل انهدام توسط Garbage collection جمع‌آوری می‌شود، ایجاد می‌شود.

برای آگاهی از اطلاعات بیشتر درباره پشتیبانی از ریسمان‌سازی (Threading)، بخش «ریسمان‌سازی در .NET Compact Framework» را ملاحظه نمایید.

بازه‌ها و وقفه‌های زمانی

خاصیت **Now** مقداری برمی‌گرداند که به طور اخص برای ثانیه‌هاست نه میلی ثانیه. با استفاده از خاصیت **TickCount** می‌توانید اندازه ریزتری را به دست آورید.

زمان سنج‌ها

متدهای Start و Stop برای یک شیء System.Timers.Timer پشتیبانی نمی‌شوند، اما با تنظیم خاصیت Enabled یک شیء System.Windows.Forms.Timer به **true** یا **false**، می‌توانید شروع به زمان‌سنجی کرده یا آن را متوقف نمایید.

Visual Basic My

.NET Compact Framework از مشخصه **My** ویژوال بیسیک پشتیبانی می‌کند، به جز برای اشیاء **My** زیر:

- **My.Application**
- **My.Computer**
- **My.User**
- **My.Settings**

سرویس‌های وب

مشتری (سرویس گیرنده) سرویس‌های وب اسمبلی‌هایی را که توسط wsdl.exe تولید می‌شوند به طور مستقیم اجرا می‌کند.

از localhost برای ایجاد یک سرویس وب روی دستگاه استفاده نکنید، زیرا localhost به دستگاهی اشاره می‌کند که در حال اجرای برنامه است. در عوض، شما باید یا از اسم ماشین استفاده کنید یا از آدرس IP آن.

XML

به علت ملاحظات اندازه، .NET Compact Framework از ارزیابی الگوی XML پشتیبانی نمی‌کند. .NET Compact Framework از XML Document Object Model (DOM) پشتیبانی نمی‌کند.

کلاس‌های پشتیبانی شده در .NET Compact Framework

.NET Compact Framework حدود ۳۰ درصد از کلاس‌ها و فضاهای نامی .NET Framework کامل را پشتیبانی می‌کند، و اندازه آن تقریباً ۸ درصد اندازه .NET Framework کامل است. برای کم نگه داشتن اندازه، تنها با اهمیت‌ترین کلاس‌ها و اعضاء پشتیبانی می‌شوند.

تشخیص پشتیبانی

برای آگاهی از اطلاعات بیشتر درباره چگونگی استفاده از کتابخانه کلاس برای تعیین پشتیبانی .NET Compact Framework و نسخه‌پردازی، بخش «چگونگی مشخص کردن اعضای پشتیبانی شده در کتابخانه کلاس» را ملاحظه کنید.

کارکردهای پشتیبانی نشده

.NET Compact Framework از تکنولوژی‌های زیر پشتیبانی نمی‌کند:

- کارکرد سرور
- ASP.NET
- هدایت از راه دور (Remoting)
- منتشر کردن انعکاس (Reflection Emit)
- توسعه برنامه در C++

- توسعه برنامه در #J و JSL

NET Compact Framework. فعلاً از امنیت دسترسی به کد پشتیبانی نمی‌کند.

کلاس‌های انحصاری .NET Compact Framework

کلاس‌ها و فضاهای نامی زیر تنها در NET Compact Framework. در دسترس هستند.

نوع یا فضای نام
Microsoft.WindowsMobile.DirectX
Microsoft.WindowsMobile.DirectX.Direct3D
Microsoft.WindowsCE.Forms
IrDAEndPoint IrDACharacterSet IrDAClient IrDADeviceInfo IrDAHints IrDAListener
System.Data.SqlServerCe
Microsoft.ServiceModel.Channels.Mail
Microsoft.ServiceModel.Channels.Mail.WindowsMobile

نکته:

به طور پیش فرض، مستندات SQL Server Compact 3.5 به طور موضعی نصب نشده‌اند. برای دانلود کتاب‌های آنلاین SQL Server Compact 3.5، به لینک http://go.microsoft.com/fwlink/?LinkId=66522 مراجعه نمایید.

مدل توسعه دهنده .NET Compact Framework

.NET Compact Framework مدلی را برای توسعه دهندگان ارائه می‌کند تا آن را به منظور ایجاد برنامه‌ها و اجزایی که می‌توانند یا دامنه وسیعی از دستگاهها و پلت‌فرم‌ها یا دسته به خصوصی از دستگاهها را هدف‌گیری کنند، مورد استفاده قرار دهند.

نکته:

.NET Compact Framework برای توسعه برنامه‌های مشتری روی دستگاه‌های هوشمند تدارک دیده شده است. چنان چه می‌خواهید برنامه‌های سرور مبتنی بر وبی را برای مشتری‌های موبایل توسعه دهید، بخش «ایجاد برنامه‌های ASP.NET Mobile Web» را ملاحظه نمایید.

ساختن بر روی اجزای هسته‌ای

اجزای هسته‌ای .NET Compact Framework متشکل از CLR به اضافه اجزای زیر است:

- زیرمجموعه‌ای غنی از کتابخانه کلاس .NET Framework کامل، شامل پشتیبانی برای سرویس‌های وب XML، Windows Forms، داده‌ها، ترسیم و غیره.
- کلاس‌هایی که از ارتباطات اینفرارد پشتیبانی می‌کنند.
- کنترل‌ها و اجزای واقع در فضای نام Microsoft.WindowsCE.Forms.
- انواعی که از برنامه‌نویسی Mobile Direct3d مدیریت شده در .NET Compact Framework پشتیبانی می‌کنند.
- اجزای زبانی Visual C# و Visual Basic 2008.

این اجزای هسته‌ای یک بنیان اصلی را برای دستگاه‌ها و پلت‌فرم‌های لیست شده در بخش «دستگاه‌ها و پلت‌فرم‌های پشتیبانی شده توسط .NET Compact Framework» ارائه می‌دهند.

مایکروسافت و اشخاص ثالث می‌توانند کامپوننت‌های اختیاری را بسازند که کارکردهای .NET Compact Framework، از قبیل پایگاه داده، پیغام‌رسانی و اجزای رابط کاربری ویژه را بسط دهند. برای نمونه، SQL Server نسخه Compact، پشتیبانی لازم برای پایگاه داده را در اختیار می‌گذارد.

ایجاد کامپوننت‌های بیشتر

سازمان شما می‌تواند اجزایی را به صورت add-onها برای ویژوال استودیو ۲۰۰۸ توسعه دهد به طوری که توسعه دهندگان می‌توانند برنامه‌های پرباری را با استفاده اجزای شما بسازند:

- مشخص کردن اجزای هسته‌ای .NET Compact Framework که جزء شما به آنها وابسته است. این عمل مشخص می‌کند که آیا کامپوننت شما نامزد اجرا شدن در میان گستره دستگاه‌های .NET Compact Framework است یا حضور عاملیتی را برای دسته به خصوصی از دستگاه‌ها مدنظر دارد.
- به طور مقتضی انتخاب یک اسم فضای نام درست به منظور کپسوله کردن اجزایتان. انتخاب اسمی فضای نام درست و شایسته برای دادن یک تجربه مستحکم و اجتناب از تصادمات فضای نام با توسعه دهندگان اجزای دیگر، مهم است.
- گذاشتن جزءتان در معرض دید توسعه‌دهندگان در زمان طراحی.
- بسته‌بندی کردن این آیتم‌ها یا در یک برنامه نصب مستقل یا در یکی که به صورت یک add-on در SDE نصب می‌شود.

هنگام ایجاد کتابخانه‌های کلاس برای یک جزء، از قالب فضای نام زیر استفاده کنید:

[Company Name].[Device Type].[Technology]

برای نمونه، یک جزء مایکروسافت فرضی برای فعل و انفعال متقابل با Today Screen یک Pocket PC، فضای نام زیر را خواهد داشت:

Microsoft.PocketPC.TodayScreen

این فضای نام دارای "Microsoft" به عنوان شرکت عرضه کننده جزء، "PocketPC" به عنوان نوع دستگاه خاصی که هدف‌گیری می‌شود و "TodayScreen" به عنوان تکنولوژی که توسط جزء در معرض دید گذاشته می‌شود، است.

مایکروسافت و اشخاص ثالث می‌توانند اجزای اختیاری را بسازند که کارکردهای .NET Compact Framework، از قبیل پایگاه داده، پیغام‌رسانی و اجزای رابط کاربری ویژه را بسط دهند.

اجزای هسته‌ای و کارکردهای بسط یافته

اجزای زیر به عنوان هسته‌ای برای .NET Compact Framework در نظر گرفته می‌شوند:

- CLR
- زیرمجموعه پرباری از کلاس‌های .NET Framework
- کلاس‌های مختص .NET Compact Framework. شامل کلاس‌هایی برای استفاده SQL Server CE
- اجزای زبان Visual Basic 2008
- اجزای زبان Visual C#

اجزای اختیاری امکان‌پذیر زیر می‌توانند کارکردهای .NET Compact Framework را بسط و گسترش دهند:

- ملحقات Pocket PC
- پایگاه‌های داده طرف ثالث
- پیغام رسانی از یک طرف ثالث

.NET Compact Framework در تمامی دستگاههای هوشمند مایکروسافت، شامل دستگاههای Pocket PC، Pocket PC Phone Edition، دستگاههای Smartphone و دیگر دستگاههای توانمند شده با Windows Embedded CE به صورت یک جزء سیستم عامل در دسترس است.

لیست پشتیبانی

جدول زیر دستگاهها و پلت‌فرم‌های حمایت شده توسط .NET Compact Framework را لیست می‌کند.

نسخه .NET Compact Framework	دستگاه	پلت‌فرم
1.0	Pocket PC	Pocket PC 2000, Pocket PC 2002, Windows Mobile 2003 for Pocket PC, Windows Mobile 2003 for Pocket PC SE, Windows Mobile 5.0 software for Pocket PC
1.0	Smartphone	Windows Mobile 2003 for Smartphone Windows Mobile 5.0 software for Smartphone
1.0	Other Windows Embedded CE devices	Windows CE 4.1 Windows CE 4.2 Windows CE 5.0
2.0	Pocket PC	Windows Mobile 2003 for Pocket PC, Windows Mobile 2003 for Pocket PC SE, Windows Mobile 5.0 software for Pocket PC
2.0	Smartphone	Windows Mobile 5.0 software for Smartphone
2.0	Other Windows Embedded CE devices	Windows CE 4.2 Windows CE 5.0 Windows Embedded CE 6.0
3.5	Pocket PC	Windows Mobile 2003 for Pocket PC, Windows Mobile 2003 for Pocket PC SE, Windows Mobile 5.0 software for Pocket PC
3.5	Smartphone	Windows Mobile 5.0 software for Smartphone
3.5	Other Windows Embedded CE devices	Windows Embedded CE 6.0

نکته:

.NET Compact Framework نسخه 2.0 نیازمند Windows Mobile نسخه 5.0 در حال اجرا روی دستگاه است. برای آگاهی از اطلاعات بیشتر درباره نصب SDKهای Windows Mobile 5.0 برای Pocket PC و Smartphone، بخش «منابع خارجی برای .NET Compact Framework» را ملاحظه نمایید.

چنان چه .NET Compact Framework نسخه 1.0 از قبل در حافظه فقط خواندنی (ROM) حضور نداشته باشد، می‌تواند روی دستگاههای Pocket PC 2000، 2002، 2003 و 2003 SE، در حافظه با دسترسی تصادفی (RAM) نصب شود. چنان چه .NET Compact Framework نسخه 2.0 از قبل در ROM حضور نداشته باشد، می‌تواند در RAM یا در انباره پایدار روی Pocket PC 2003، Windows CE 5.0، و Windows Mobile 5.0 برای Pocket PCها و Smartphoneها نصب شود. .NET Compact Framework نسخه 3.5 فعلاً در ROM در دسترس نیست، اما می‌تواند در RAM بر اساس دستگاههای حمایت شده که در جدول پیشین نشان داده شدند، نصب شود.

نسخه های ROM

ابزار Windows CE Platform Builder، OEMها را قادر می‌سازد تا .NET Compact Framework را به صورت یک جزء سیستم عامل یکپارچه در میان یک دستگاه توانمند شده با Windows Embedded CE جاسازی کند. بروزرسانی‌های ماهانه و سالانه برای Windows Embedded CE Platform اغلب اوقات شامل بروزرسانی‌های لازم برای .NET Compact Framework نیز است. Windows CE نسخه‌های 4.2، 5.0 و 6.0 از تعبیه .NET Compact Framework نسخه 2.0 یا 3.5 در ROM پشتیبانی می‌کنند.

برای دستگاههای Windows Mobile، نسخه .NET Compact Framework نصب شده در ROM به پلت‌فرم ویژه‌ای گره خورده است. با این حال، در برخی از موارد، OEMها می‌توانند مابین دو نسخه انتخاب انجام دهند. جدول زیر نسخه .NET Compact Framework را که در ROM نصب می‌شود، بر اساس نسخه پلت‌فرم نمایش می‌دهد.

نسخه .NET Compact Framework	پلت‌فرم
1.0 SP2	Windows Mobile 2003 for Pocket PC
1.0 SP3	Windows Mobile 2003 for Pocket PC SE
1.0 SP3	Windows Mobile 5.0

Windows Mobile 5.0 (build 15096.3.0)	1.0 SP3 or 2.0 (مشخص شده توسط OEM)
Windows Mobile 5.0 (build 15341.4.0)	1.0 SP3 or 2.0 SP1 (مشخص شده توسط OEM)
Windows Mobile 6.0	2.0 SP2

ارتباطات اینفرارد

.NET Compact Framework کلاس‌هایی را به منظور توسعه برنامه‌های ارتباطی با اینفرارد برای دستگاههای هوشمند و کامپیوترهای شخصی ارائه می‌دهد. این کلاس‌ها جزو ضمایم فضای نام System.Net.Sockets هستند و برنامه‌نویسی سوکتی را پیاده‌سازی می‌کنند که با استانداردهای تنظیم شده توسط مشخصات Infrared Data Association (IrDA) مطابقت دارد.

استفاده از کلاس‌های IrDA

.NET Framework کامل دارای کلاس‌های IrDA نیست. به منظور برپایی ارتباطات IrDA مابین یک کامپیوتر شخصی و یک دستگاه، بایستی با استفاده از احضار پلت‌فرم در سمت کامپیوتر شخصی، سوکت ویندوز را احضار کنید.

نکته:

.NET Compact Framework تنها از استاندارد IrDA پشتیبانی می‌کند. برای دسترسی به دیگر قابلیت‌های اینفرارد روی یک دستگاه، بایستی از احضار پلت‌فرم استفاده کنید. .NET Compact Framework از دستگاه‌هایی که با Windows CE توانمند شده‌اند با یک پورت IrDA، مانند ماوس‌ها، پرینترها، Pocket PCها و دیگر PDAها پشتیبانی می‌کند.

IrDA مجموعه پروتکلی را تعریف می‌کند که طراحی شده‌اند تا از انتقال داده اینفرارد به صورت نقطه به نقطه مابین دو دستگاه بی‌سیم روی دامنه‌ای با برد کم پشتیبانی کند. IrDA ارتباطی جفتی را تطبیق می‌دهد جایی که دستگاه‌های متعدد می‌توانند ارتباطاتی را با میزبان (Host) یکسانی برقرار کنند.

شما می‌توانید عاملیت (کارکرد) سرور و مشتری را پیاده‌سازی نمایید. مشتری، دستگاهی است که ارتباط را راه‌اندازی می‌کند. بعد از برقراری یک ارتباط، داده‌ها می‌توانند به طور قابل اطمینانی، معاوضه و مبادله شوند. از آن جایی که یک سرور به نیازمندی‌های پشته (Stack) اضافی نیاز دارد، سرور نوعاً یک کامپیوتر شخصی است.

کلاس‌های IrDA در لایه Information Access Service (IAS) پشته پروتکل IrDA عمل می‌کنند. یک لایه IAS می‌تواند یک یا هردو مؤلفه زیر را دارا باشد:

- Server. دربردارنده مجموعه‌ای از اشیاء است که سرویس‌ها و برنامه‌های در دسترس برای ارتباطات وارده به آن سرور را توصیف می‌کند.

> **نکته:**
>
> در این نسخه منتشر شده از .NET Compact Framework. هیچ گونه کلاسی برای جزء سرور وجود ندارد.

- Client. پرس‌وجوهای اکتشافی را روی مؤلفه سرور انجام می‌دهد تا فهرستی از تمامی ارتباطات در دسترس را به دست آورد.

جدول زیر توابع انجام شده توسط کلاس‌های IrDA را در .NET Compact Framework. جمع‌بندی می‌کند.

کلاس	توضیح
IrDACharacterSet	این برشمارش (ریزفهرست یا Enumeration) مجموعه کاراکتر پشتیبانی شده توسط دستگاه IrDA که یافت شده است را توصیف می‌کند.
IrDAClient	دسترسی به مشتری را در اختیار قرار می‌دهد به طوری که شما می‌توانید یک ارتباط دلخواه را تعیین کرده، آن را باز کنید و داده‌ها را ارسال و دریافت نمایید.

IrDADeviceInfo	اطلاعاتی را در مورد ارتباطات در دسترس روی یک سرور به دست آمده توسط یک پرس‌وجوی اکتشافی از سوی مشتری، در اختیار قرار می‌دهد.
IrDAEndPoint	برای ایجاد یک ارتباط با یک سرور و کسب اطلاعات پورت اینفرارد، تدارک لازم را انجام می‌دهد.
IrDAHints	این ریزفهرست مقادیری را که نوع دستگاه یا ارتباط، مانند یک فکس را نشان می‌دهند، لیست می‌کند.
IrDAListener	یک سوکت را در وضعیت شنود به منظور مانیتورینگ ارتباطات در دسترس برای یک دستگاه مشخص، جای می‌دهد. گوش دهنده تا زمان فراخوانی متد Start شنود نمی‌کند.

چگونگی انجام یک انتقال فایل با Infrared

.NET Compact Framework کلاس‌هایی را به منظور ارتباطات مابین دستگاهها در اختیار می‌گذارد. این مثال چگونگی ارسال و دریافت فایل‌ها مابین دستگاه‌ها با استفاده از ارتباطات اینفرارد را تشریح می‌کند. برای پیاده‌سازی و اجرای این مثال نیاز به دو Pocket PC دارید، یکی برای ارسال فایل و دیگری برای دریافت آن.

این مثال نمونه‌ای از IrDAClient را ایجاد کرده و از متد DiscoverDevices آن استفاده می‌کند تا دستگاههای اینفرارد را در میان حوزه دسترسی (برد) Infrared پیدا کند. این متد آرایه‌ای از اشیاء IrDADeviceInfo برمی‌گرداند که اطلاعاتی را درمورد هر دستگاه تأمین می‌کنند.

این مثال کدی را برای ارسال و دریافت یک فایل در اختیار قرار می‌دهد، که می‌توانند با یک دکمه Send و Receive نشان داده شوند. دست کم شما نیاز خواهید داشت تا یک برنامه Send را برای یک دستگاه و یک برنامه Receive را برای دستگاه دیگر ایجاد کنید.

دکمه Send تنها یک فایل را به دستگاهی ارسال می‌کند که در حال گوش دادن به درخواست‌هایی که یک فایل ارسال خواهد شد، بوده است. از این رو، قبل از این دکمه Send روی دستگاه ارسال کننده انتخاب شود، بایستی دکمه Receive روی دستگاه دریافت کننده انتخاب شود. برای این منظور کارهای زیر انجام خواهند شد:

- به دست آوردن جریانی از فایل را برای ارسال.
- ایجاد نمونه‌ای از IrDAClient با استفاده از نام سرویس تعیین شده برای این برنامه. ارتباطات اینفرارد با مشخص کردن یک نام سرویس ساخته می‌شوند، که این نام می‌تواند هر مقداری باشد به شرطی که دستگاههای شرکت کننده به نام یکسانی اشاره کنند. در این مثال نام سرویس "IrDATest" است.
- خواندن جریانی فایل به میان جریان IrDAClient که فایل را ارسال می‌کند.

دکمه Receive نمونه‌ای از IrDAListener را ایجاد می‌کند تا دستگاهی را با همان نام سرویس به صورت نمونه IrDAClient روی دستگاه ارسال کننده شنود کند.

کارهای زیر انجام می‌پذیرند:

- ایجاد یک جریان به منظور نوشتن محتویات منتقل شده به یک فایل دریافت کننده واقع در پوشه **My Documents**.
- ایجاد یک نمونه IrDAEndPoint با ID دستگاه و نام سرویس برای دستگاه ارسال کننده.
- ایجاد یک نمونه IrDAListener از نمونه IrDAEndPoint و راه‌اندازی سرویس شنود.
- ایجاد یک نمونه IrDAClient از نمونه IrDAListener با استفاده از متد AcceptIrDAClient.
- خواندن جریان متضمن نمونه IrDAClient، که حاوی داده‌های فایل منتقل شده است.
- نوشتن آن جریان فایلی به میان جریان مربوط به فایل Receive.txt.

برای ایجاد برنامه‌ها

۱. یک برنامه Pocket PC برای دستگاه ارسال کننده ایجاد کرده و یک دکمه به فرم بیفزایید. نام دکمه را Send بگذارید.

۲. در پوشه **My Documents** فایلی به نام Send.txt را ایجاد کنید.

۳. کد زیر را برای رویداد Click دکمه Send اضافه کنید.

Visual Basic کد نمونه:

```
' Align the infrared ports of the devices.
' Click the Receive button first, then click Send.
Private Sub SendButton_Click(sender As Object, e As System.EventArgs) _
    Handles SendButton.Click

    Dim irClient As New IrDAClient()
    Dim irServiceName As String = "IrDATest"
    Dim irDevices() As IrDADeviceInfo
    Dim buffersize As Integer = 256

    ' Create a collection of devices to discover.
    irDevices = irClient.DiscoverDevices(2)

    ' Show the name of the first device found.
    If irDevices.Length = 0 Then
        MsgBox("No remote infrared devices found.")
        Return
    End If

    Try
        Dim irEndP As New IrDAEndPoint(irDevices(0).DeviceID, _
            irServiceName)
        Dim irListen As New IrDAListener(irEndP)
        irListen.Start()
        irClient = irListen.AcceptIrDAClient()
        MsgBox("Connected!")

    Catch exSoc As SocketException
         MsgBox("Couldn't listen on service " & irServiceName & ": " _
            & exSoc.ErrorCode)
    End Try

    ' Open a file to send and get its stream.
    Dim fs As Stream
    Try
        fs = New FileStream(".\My Documents\send.txt", FileMode.Open)
    Catch exFile As Exception
        MsgBox("Cannot open " & exFile.ToString())
        Return
    End Try
```

```
    ' Get the underlying stream of the client.
    Dim baseStream As Stream = irClient.GetStream()

    ' Get the size of the file to send
    ' and write its size to the stream.
    Dim length As Byte() = BitConverter.GetBytes(fs.Length)
    baseStream.Write(length, 0, length.Length)

    ' Create a buffer for reading the file.
    Dim buffer(buffersize) As Byte

    Dim fileLength As Integer = CInt(fs.Length)

    Try
        ' Read the file stream into the base stream.
        While fileLength > 0
            Dim numRead As Int64 = fs.Read(buffer, 0, buffer.Length)
            baseStream.Write(buffer, 0, numRead)
            fileLength -= numRead
        End While
        MsgBox("File sent")
    Catch exSend As Exception
        MsgBox(exSend.Message)
    End Try

    fs.Close()
    baseStream.Close()
    irClient.Close()
End Sub
```

C# کد نمونه:

```
// Align the infrared ports of the devices.
// Click the Receive button first, then click Send.
private void SendButton_Click(object sender, System.EventArgs e)
{
    IrDAClient irClient = new IrDAClient();
    string irServiceName = "IrDATest";
    IrDADeviceInfo[] irDevices;
    int buffersize = 256;

    // Create a collection of devices to discover.
    irDevices = irClient.DiscoverDevices(2);

    // Show the name of the first device found.
    if ((irDevices.Length == 0))
    {
        MessageBox.Show("No remote infrared devices found.");
        return;
    }
    try
    {
        IrDAEndPoint irEndP =
            new IrDAEndPoint(irDevices[0].DeviceID, irServiceName);
        IrDAListener irListen = new IrDAListener(irEndP);
        irListen.Start();
```

```
        irClient = irListen.AcceptIrDAClient();
        MessageBox.Show("Connected!");
    }
    catch (SocketException exSoc)
    {
        MessageBox.Show(("Couldn\'t listen on service "
                        + (irServiceName + (": " + exSoc.ErrorCode))));
    }

    // Open a file to send and get its stream.
    Stream fs;
    try
    {
        fs = new FileStream(".\\My Documents\\send.txt", FileMode.Open);
    }
    catch (Exception exFile)
    {
        MessageBox.Show(("Cannot open " + exFile.ToString()));
        return;
    }

    // Get the underlying stream of the client.
    Stream baseStream = irClient.GetStream();

    // Get the size of the file to send
    // and write its size to the stream.
    byte[] length = BitConverter.GetBytes(fs.Length);
    baseStream.Write(length, 0, length.Length);

    // Create a buffer for reading the file.
    byte[] buffer = new byte[buffersize];
    int fileLength = (int) fs.Length;
    try
    {
        // Read the file stream into the base stream.
        while ((fileLength > 0))
        {
            Int64 numRead = fs.Read(buffer, 0, buffer.Length);
            baseStream.Write(buffer, 0, Convert.ToInt32(numRead));
            fileLength = (fileLength - Convert.ToInt32(numRead));
        }
        MessageBox.Show("File sent");
    }
    catch (Exception exSend)
    {
        MessageBox.Show(exSend.Message);
    }
    fs.Close();
    baseStream.Close();
    irClient.Close();
}
```

۴. یک برنامه Pocket PC برای دستگاه دریافت کننده ایجاد کرده و یک دکمه به فرم بیفزایید. نام دکمه را Receive بگذارید.

۵. کد زیر را به رویداد Click دکمه Receive اضافه کنید.

Visual Basic کد نمونه:

```
' Align the infrared ports of the devices.
' Click the Receive button first, then click Send.
Private Sub ReceiveButton_Click(sender As Object, e As System.EventArgs) _
    Handles ReceiveButton.Click

    Dim irDevices() As IrDADeviceInfo
    Dim irClient As New IrDAClient()
    Dim irServiceName As String = "IrDATest"

    Dim buffersize As Integer = 256

    ' Create a collection for discovering up to
    ' three devices, although only one is needed.
    irDevices = irClient.DiscoverDevices(2)

    ' Cancel if no devices are found.
    If irDevices.Length = 0 Then
        MsgBox("No remote infrared devices found.")
        Return
    End If

    ' Connect to the first IrDA device
    Dim irEndP As New IrDAEndPoint(irDevices(0).DeviceID, irServiceName)
    irClient.Connect(irEndP)

    ' Create a stream for writing a Pocket Word file.
    Dim writeStream As Stream
    Try
        writeStream = New FileStream(".\My Documents\receive.txt", _
            FileMode.OpenOrCreate)
    Catch
        MsgBox("Cannot open file for writing.")
        Return
    End Try

    ' Get the underlying stream of the client.
    Dim baseStream As Stream = irClient.GetStream()

    ' Create a buffer for reading the file.
    Dim buffer(buffersize) As Byte

    Dim numToRead, numRead As Int64

    ' Read the file into a stream 8 bytes at a time.
    ' Because the stream does not support seek operations,
    ' its length cannot be determined.
    numToRead = 8

    Try
        While numToRead > 0
            numRead = baseStream.Read(buffer, 0, numToRead)
            numToRead -= numRead
```

```
        End While
    Catch exReadIn As Exception
        MsgBox("Read in: " & exReadIn.Message)
    End Try

    ' Get the size of the buffer to show
    ' the number of bytes to write to the file.
    numToRead = BitConverter.ToInt64(buffer, 0)

    Try
        ' Write the stream to the file until
        ' there are no more bytes to read.
        While numToRead > 0
            numRead = baseStream.Read(buffer, 0, buffer.Length)
            numToRead -= numRead
            writeStream.Write(buffer, 0, numRead)
        End While
        writeStream.Close()
        MsgBox("File received.")
    Catch exWriteOut As Exception
        MsgBox("Write out: " & exWriteOut.Message)
    End Try

     baseStream.Close()
     irClient.Close()
End Sub
```

C# کد نمونه:

```
// Align the infrared ports of the devices.
// Click the Receive button first, then click Send.
private void ReceiveButton_Click(object sender, System.EventArgs e)
{
    IrDADeviceInfo[] irDevices;
    IrDAClient irClient = new IrDAClient();
    string irServiceName = "IrDATest";
    int buffersize = 256;

    // Create a collection for discovering up to
    // three devices, although only one is needed.
    irDevices = irClient.DiscoverDevices(2);

    // Cancel if no devices are found.
    if ((irDevices.Length == 0))
    {
        MessageBox.Show("No remote infrared devices found.");
        return;
    }

    // Connect to the first IrDA device
    IrDAEndPoint irEndP =
        new IrDAEndPoint(irDevices[0].DeviceID, irServiceName);
    irClient.Connect(irEndP);

    // Create a stream for writing a Pocket Word file.
    Stream writeStream;
```

```
    try
    {
        writeStream = new FileStream(".\\My Documents\\receive.txt",
            FileMode.OpenOrCreate);
    }
    catch (Exception Ex)
    {
        MessageBox.Show("Cannot open file for writing. " + Ex.Message);
        return;
    }

    // Get the underlying stream of the client.
    Stream baseStream = irClient.GetStream();

    // Create a buffer for reading the file.
    byte[] buffer = new byte[buffersize];
    Int64 numToRead;
    Int64 numRead;

    // Read the file into a stream 8 bytes at a time.
    // Because the stream does not support seek operations, its
    // length cannot be determined.
    numToRead = 8;
    try
    {
        while ((numToRead > 0))
        {
            numRead = baseStream.Read(buffer, 0,
                Convert.ToInt32(numToRead));
            numToRead = (numToRead - numRead);
        }
    }
    catch (Exception exReadIn) {
        MessageBox.Show(("Read in: " + exReadIn.Message));
    }

    // Get the size of the buffer to show
    // the number of bytes to write to the file.
    numToRead = BitConverter.ToInt64(buffer, 0);
    try
    {
        // Write the stream to the file until
        // there are no more bytes to read.
        while ((numToRead > 0)) {
            numRead = baseStream.Read(buffer, 0, buffer.Length);
            numToRead = (numToRead - numRead);
            writeStream.Write(buffer, 0, Convert.ToInt32(numRead));
        }
        writeStream.Close();
        MessageBox.Show("File received.");
    }
    catch (Exception exWriteOut)
    {
        MessageBox.Show(("Write out: " + exWriteOut.Message));
    }
    baseStream.Close();
    irClient.Close();
}
```

برای اجرای برنامه‌ها

۱. برنامه‌ها را به دستگاه‌ها توزیع کرده و آنها را راه‌اندازی کنید.

۲. پورت‌های اینفرارد دستگاه‌ها را تنظیم کنید.

۳. دکمه Receive را روی دستگاه دریافت کننده به چنگ آورده و کلیک کنید.

۴. دکمه Send را روی دستگاه ارسال کننده به چنگ آورده و کلیک کنید.

۵. بررسی کنید تا ببینید که آیا Receive.txt در پوشه **My Documents** ایجاد شده است یا نه.

کامپایل کد

این مثال نیازمند ارجاعات به فضاهای نام زیر است:

- System
- System.Net
- System.Net.Sockets
- System.IO
- System.Windows.Forms

شبکه‌سازی و ارتباط در .NET Compact Framework

.NET Compact Framework برای شبکه‌سازی API با تراز سوکت و تجریدهای با تراز بالاتر، مانند HTTP، Domain Name System (DNS) و درخواست‌ها و پاسخ‌های وب را عرضه می‌کند. اتصال روی Infrared Data Association (IrDA) و ترانسپورت‌های TCP/IP از طریق سوکت‌ها ارائه می‌شوند.

برنامه‌نویسی شبکه در .NET Compact Framework

.NET Compact Framework پشتیبانی توکاری را برای سرویس‌های وب XML در اختیار قرار می‌دهد و پشتیبانی پروتکل و عاملیتها (کارکردها) زیر را ارائه می‌دهد:

- پروتکل‌های مبتنی بر HTTP.
- تعیین اعتبار NTLM.

- محتویات XML با رمزگذاری SOAP. این پشتیبانی شامل ارسال مجموعه داده‌های ADO.NET است.
- متدهای Web Request و Web Response که می‌توانند پیغام‌های HTTP SOAP را ارسال کرده و پیغام‌های SOAP را در پاسخ دریافت کنند.
- متدها و کتابخانه‌های SOAP که می‌توانند فراخوان‌های متد و اشیای دلبخواهی را در میان پیغام‌های XML SOAP سریالی کرده و از آنها بزداید.

درخواست‌های HTTP

آیتم‌های زیر مربوط به ارسال و دریافت درخواست‌های HTTP هستند:

- هنگام استفاده از نمونه‌ساز، localhost را برای نام سرور به کار نبرید. نام کامپیوتر یا آدرس IP کامپیوتر توسعه دهنده خود را که میزبانی سرویس وب را می‌کند، مشخص کنید.

نمونه‌ساز، مانند یک دستگاه، دارای آدرس IP مال خودش است. استفاده از localhost به نمونه‌ساز دستور می‌دهد تا به جای سرویس وب میزبان شده توسط محیط توسعه شما یا کامپیوتر رومیزی دیگری، از خودش استفاده کند تا به سرویس وب متصل شود.

برای مثال، به جای آدرس زیر:

http://localhost/myWebService/Service1.asmx

آدرس زیر را مشخص کنید:

http://ComputerName/myWebService/Service1.asmx

- هنگام انجام یک درخواست HTTP با استفاده از HttpWebRequest، دستگاه چنان چه اتصالی در دسترس نباشد، یک ارتباط شبکه جدید را راه‌اندازی می‌کند. از این رو، انجام یک درخواست HTTP تنها برای مشخص کردن یک اتصال در دسترس، می‌تواند باعث شود که دستگاه تلاش کند تا یک اتصال را راه‌اندازی کند، برای مثال یک اتصال GPRS.

- .NET Compact Framework اطلاعات پراکسی را در خاصیت GlobalProxySelection.Select ذخیره نمی‌کند، اما چنان چه مقداری را در کد خود تعیین کنید، از این خاصیت برای ارتباطات HTTP استفاده خواهد کرد.

- برای وصل شدن به اینترنت، ممکن است نیازمند مشخص کردن تنظیمات پراکسی محلی خود باشید. کد زیر تنظیم پراکسی را برای پورت ۸۰ نمایش می‌دهد:

کد نمونه:

```
System.Net.GlobalProxySelection.Select = new WebProxy("http://myproxy:80");
```

- چنان چه AllowWriteStreamBuffering را برابر **false** قرار دهید، داده‌ها در حافظه بافر (میانگیری) نخواهند شد و هر گونه درخواست برای تعیین اعتبار یا تعیین مسیر توسط سرور وب را پشتیبانی نخواهد کرد.

- به منظور حصول اطمینان از اعمال موفقیت‌آمیز، اطلاعات مسیر مطلق را مشخص کنید.

به رفتار زیر Windows CE در تجزیه و تحلیل مشخصات فایل نسبی توجه کنید:

`file://myfile` به صورت `\\myfile` تجزیه می‌شود.

`file:///myfile` به صورت `\myfile` در دایرکتوری ریشه تجزیه می‌شود.

- این یک مسئله شناخته شده است که هنگامی که بیشتر از ۵۰ پروتکل شبکه نصب شده روی ماشین جاری وجود داشته باشد، متد Dns.GetHostName برای .NET Framework. یک استثناء پرتاب می‌کند.

برای کار پیرامون این مشکل، پروتکل‌های شبکه‌ای را که واقعاً مورد نیاز نیستند، Uninstall کنید. یک روش برای انجام این کار این است که برای حذف آداپتورهای شبکه نامستعمل، از Windows Device Manager استفاده کنید. روش دیگر این است که برنامه‌هایی را که پروتکل‌ها را نصب کرده‌اند، Uninstall نمایید.

ارتباطات موبایل ایمن

دو وسیله عمده برای ارتباطات موبایل ایمن وجود دارد:

- تصدیق اعتبار HTTP

.NET Compact Framework از تصدیق اعتبار Basic و Digest پشتیبانی می‌کند. این مکانیزم‌های تصدیق اعتبار ساده هستند، و امنیت و سبک و سنگین کردن آنها نسبتاً شناخته شده است— یکی این که سرویس وب محدود به یک انقیاد HTTP است.

.NET Compact Framework نسخه 2.0 از سرورهای در حال اجرای NTLM یا Kerberos («تصدیق اعتبار یکپارچه ویندوز»)[1]، که نیاز ندارد که هیچ تغییری کدی به تصدیق اعتبار Basic یا Digest اعمال شود، پشتیبانی می‌کند.

- هدرهای امنیتی سفارشی

.NET Compact Framework فعلاً از Web Services Security (WS-Security) و Web Service Enhancements (WSE) پشتیبانی نمی‌کند.

علاوه بر این، چه با استفاده از HTTP یا یک هدر سفارشی تصدیق اعتبار کنید، می‌توانید از SSL به منظور افزایش امنیت استفاده کنید. تصدیق اعتبار Basic، نام و کلمه عبور را در متنی واضح (clear text) ارسال می‌کند، پس به طور اخص ایمن نیست مگر این که از میان SSL اجرا شود. با این حال، هنگامی که در اجتماع با SSL به کار برده شود، به طور نسبتاً خوبی ایمن خواهد بود، با تنها مشکل عمده‌ای که به طور ناگهانی در حال فاش کردن اعتبارنامه‌ها برای سرور هدف پیش می‌آید.

نکته .NET Compact Framework از تصدیق اعتبار سمت مشتری با استفاده از تأییدیه‌های X509 پشتیبانی نمی‌کند.

[1] Windows Integrated Authentication

ملاحظات مربوط به طول محتویات

هنگام ارسال یک درخواست وب HTTP از محتوای جریان‌داری شده با استفاده از پروتکل POST، بایستی یک طول محتوا را تعیین کنید. با فرض این که SendChunked برابر **false** باشد و Method = POST، مقداری را برای ContentLength مشخص کنید.

برخلاف .NET Framework کامل، .NET Compact Framework داده‌ها را از پیش میانگیری نمی‌کند تا ملاحظات مربوط به محدودیت حافظه را اصلاح کند. برای حصول اطمینان از انجام عمل میانگیری (Buffering)، SendChunked را برابر **false** قرار دهید.

یک درخواست با طول محتوای صفر باعث بروز یک استثنای ObjectDisposedException می‌شود اگر به درستی وصول و بسته شده نباشد. به منظور مدیریت درخواست‌های با طول محتوای صفر، بایستی به طور صریح متد GetRequestStream را فراخوانده و پس از آن متد Close را روی جریان برگشت داده شده بدون فراخوانی متد Write فراخوان نمایید، همان طور که در کد نمونه زیر نشان داده شده است.

C# کد نمونه:

```
private static void ZeroLengthRequest()
{
    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(
        requestUri + "?dummy=true");
    request.AllowWriteStreamBuffering = true;
    request.Credentials = CredentialCache.DefaultNetworkCredentials;
    request.Credentials = netCred;
    request.ConnectionGroupName = "mygroup";
    request.ContentLength = 0;
    request.KeepAlive = true;
    request.Method = "POST";
    request.ServicePoint.UseNagleAlgorithm = false;
    request.Timeout = System.Threading.Timeout.Infinite;
    request.UnsafeAuthenticatedConnectionSharing = true;
    Stream req = request.GetRequestStream();
    req.Close();
    using (request.GetResponse())
    {
        ...
    }
}
```

Visual Basic کد نمونه:

```
Private Shared Sub ZeroLengthRequest()
```

```
        Dim request As HttpWebRequest = _
           CType(WebRequest.Create(requestUri + "?dummy=true"), _
           HttpWebRequest)
        request.AllowWriteStreamBuffering = true
        request.Credentials = CredentialCache.DefaultNetworkCredentials
        request.Credentials = netCred
        request.ConnectionGroupName = "mygroup"
        request.ContentLength = 0
        request.KeepAlive = true
        request.Method = "POST"
        request.ServicePoint.UseNagleAlgorithm = false
        request.Timeout = System.Threading.Timeout.Infinite
        request.UnsafeAuthenticatedConnectionSharing = true
        Dim req As Stream = request.GetRequestStream
        req.Close
        request.GetResponse
End Sub
```

سوکت‌ها

فضای نام System.Net.Sockets حاوی پیاده‌سازی مدیریت شده‌ای از واسط Windows Sockets است. تمامی کلاس‌های دیگر با دسترسی شبکه در فضای نام System.Net بر بالای این پیاده‌سازی از سوکت‌ها ساخته شده است.

کلاس **Socket** برای .NET Compact Framework یک نسخه کد مدیریت شده از سرویس‌های سوکت عرضه شده توسط Winsock32 API است. در اغلب موارد، متدهای کلاس **Socket** به راحتی داده‌ها را در میان شرکای Win32 خالص‌شان مرتب کرده و هر گونه بررسی امنیتی لازم را مدیریت می‌کنند.

کلاس **Socket** از دو حالت پایه پشتیبانی می‌کند، هماهنگ (synchronous) و ناهماهنگ (asynchronous). در حالت هماهنگ، فراخوان‌ها به توابعی که اعمال شبکه (مانند Send و Receive) را انجام می‌دهند، تا زمانی که عملیات قبل از برگرداندن کنترل به برنامه فراخواننده، کامل شود، منتظر می‌ماند. در حالت ناهماهنگ، این فراخوان‌ها بلافاصله عودت داده می‌شوند.

برنامه‌نویسی سوکت

موارد زیر با برنامه‌نویسی سوکت‌ها در .NET Compact Framework ارتباط دارند. برای آگاهی از اطلاعات بیشتر درباره استفاده از سوکت‌های .NET Compact Framework.، بخش «سوکت‌ها» را ملاحظه نمایید.

- تمامی گزینه‌های سوکت روی تمامی سیستم عامل‌های دستگاه پشتیبانی نمی‌شوند.

.NET Compact Framework طراحی شده است تا قادر باشد که هر تعداد سیستم عامل را بارگیری کند، هر کدام با تراز کارکردی مال خودش. از این رو، .NET Compact Framework به طور مصنوعی میزان دسترسی گزینه‌های سوکت را بر اساس هر تراز به خصوصی از پشتیبانی از یک سیستم عامل، محدود نمی‌کند.

- سوکت‌های اولیه پشتیبانی نمی‌شوند.
- مشکلات شناخته شده‌ای با سوکت‌ها، روی Pocket PCهای اجرا کننده Windows CE 3.0 وجود دارد.

اگر یک سوکت را با داده ارسال نشده‌ای از یک فراخوان Send قبلی ببندید، داده یا زائل یا خراب خواهد شد.

اگر سوکتی را بپذیرید و سپس سوکت مقید (یا مجاور) را قبل از بستن سوکت پذیرفته شده ببندید، تا زمانی که وقفه زمانی، که حدود 4.5 دقیقه است، سپری شده باشد، نمی‌توانید به آن پورت وصل شوید.

- در برنامه های .NET Compact Framework، گزینه‌هایی که می‌آیند حمایت می‌شوند اما بدون اصلاح پشته TCP/IP کار نمی‌کنند و فعلاً برای استفاده در آینده رزرو شده‌اند: AcceptConnection، ReceiveLowWater، ReceiveTimeout، SendLowWater، SendTimeout و Type.
- عضو برشمارش ReceiveBuffer تنها برای سوکت‌های Windows CE Winsock نوع SOCK_DGRAM، که سوکت‌های datagram-oriented هستند، پشتیبانی می‌شود. اندازه بافر دریافتی پیش فرض برابر ۳۲۷۶۸ بایت است و نمی‌تواند با ReceiveBuffer تنظیم شود.

چگونگی استفاده از سوکت‌ها

.NET Compact Framework از ارتباطات شبکه‌ای مبتنی بر سوکت پشتیبانی می‌کند. برای آگاهی از ملاحظات مربوط به برنامه‌نویسی سوکت‌ها در .NET Compact Framework، بخش «برنامه‌نویسی سوکت» را ملاحظه نمایید.

این مثال نمونه‌ای از یک برنامه سرور و نمونه‌ای از یک برنامه مشتری (Client) را ایجاد کرده، و نشان می‌دهد که چگونه دو برنامه از طریق یک ارتباط مبتنی بر سوکت، با یک‌دیگر ارتباط برقرار می‌کنند. آدرس localhost برای سرور به کار برده شده است؛ از این رو، هر دو برنامه روی مشتری اجرا می‌شوند. یک نمونه از سرور، قبل از این که مشتری بتواند با آن ارتباط برقرار کند، بایستی در حال اجرا باشد.

برای ارتباط برقرار کردن از طریق یک ارتباط سوکت

۱. کلاسی به نام `Server` ایجاد کنید که `Form` را پیاده‌سازی کند، و کد زیر را به کلاس بیفزایید:

Visual Basic — کد نمونه:

```
Private Shared output As String = ""


Public Sub New()

End Sub


Public Sub createListener()
    ' Create an instance of the TcpListener class.
    Dim tcpListener As TcpListener = Nothing
    Dim ipAddress As IPAddress =
Dns.GetHostEntry("localhost").AddressList(0)
    Try
        ' Set the listener on the local IP address.
        ' and specify the port.
        tcpListener = New TcpListener(ipAddress, 13)
        tcpListener.Start()
        output = "Waiting for a connection..."
    Catch e As Exception
        output = "Error: " + e.ToString()
        MessageBox.Show(output)
    End Try
    While True
        ' Always use a Sleep call in a while(true) loop
        ' to avoid locking up your CPU.
        Thread.Sleep(10)
        ' Create a TCP socket.
        ' If you ran this server on the desktop, you could use
        ' Socket socket = tcpListener.AcceptSocket()
        ' for greater flexibility.
        Dim tcpClient As TcpClient = tcpListener.AcceptTcpClient()
        ' Read the data stream from the client.
        Dim bytes(255) As Byte
        Dim stream As NetworkStream = tcpClient.GetStream()
```

```
        stream.Read(bytes, 0, bytes.Length)
        Dim helper As New SocketHelper()
        helper.processMsg(tcpClient, stream, bytes)
    End While

End Sub


Shared Sub Main()
    Application.Run(New Server())

End Sub
```

C# کد نمونه:

```
static string output = "";

public Server()
{
}

public void createListener()
{
    // Create an instance of the TcpListener class.
    TcpListener tcpListener = null;
    IPAddress ipAddress =
Dns.GetHostEntry("localhost").AddressList[0];
    try
    {
        // Set the listener on the local IP address
        // and specify the port.
        tcpListener = new TcpListener(ipAddress, 13);
        tcpListener.Start();
        output = "Waiting for a connection...";
    }
    catch (Exception e)
    {
        output = "Error: " + e.ToString();
        MessageBox.Show(output);
    }
    while (true)
    {
        // Always use a Sleep call in a while(true) loop
        // to avoid locking up your CPU.
        Thread.Sleep(10);
        // Create a TCP socket.
        // If you ran this server on the desktop, you could use
        // Socket socket = tcpListener.AcceptSocket()
        // for greater flexibility.
```

```
        TcpClient tcpClient = tcpListener.AcceptTcpClient();
        // Read the data stream from the client.
        byte[] bytes = new byte[256];
        NetworkStream stream = tcpClient.GetStream();
        stream.Read(bytes, 0, bytes.Length);
        SocketHelper helper = new SocketHelper();
        helper.processMsg(tcpClient, stream, bytes);
    }
}

static void Main()
{
    Application.Run(new Server());
}
```

۲. در کلاس `Server`، روشی را برای راه‌اندازی سرور ارائه دهید. برای مثال، `createListener` را در رویداد Click یک دکمه فراخوان کنید.

Visual Basic کد نمونه:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
    Me.createListener()
End Sub
```

C# کد نمونه:

```
private void button1_Click(object sender, EventArgs e)
{
    this.createListener();
}
```

۳. یک کلاس به نام `SocketHelper` ایجاد کنید، و کد زیر را به کلاس بیفزایید:

Visual Basic کد نمونه:

```
Class SocketHelper
    Private mscClient As TcpClient
    Private mstrMessage As String
    Private mstrResponse As String
    Private bytesSent() As Byte

    Public Sub processMsg(ByVal client As TcpClient, ByVal stream As
NetworkStream, ByVal bytesReceived() As Byte)
```

```
        ' Handle the message received and
        ' send a response back to the client.
        mstrMessage = Encoding.ASCII.GetString(bytesReceived, 0,
bytesReceived.Length)
        mscClient = client
        mstrMessage = mstrMessage.Substring(0, 5)
        If mstrMessage.Equals("Hello") Then
            mstrResponse = "Goodbye"
        Else
            mstrResponse = "What?"
        End If
        bytesSent = Encoding.ASCII.GetBytes(mstrResponse)
        stream.Write(bytesSent, 0, bytesSent.Length)

    End Sub
End Class
```

کد نمونه: **C#**

```
class SocketHelper
{
    TcpClient mscClient;
    string mstrMessage;
    string mstrResponse;
    byte[] bytesSent;
    public void processMsg(TcpClient client, NetworkStream stream, byte[]
bytesReceived)
    {
        // Handle the message received and
        // send a response back to the client.
        mstrMessage = Encoding.ASCII.GetString(bytesReceived, 0,
bytesReceived.Length);
        mscClient = client;
        mstrMessage = mstrMessage.Substring(0, 5);
        if (mstrMessage.Equals("Hello"))
        {
            mstrResponse = "Goodbye";
        }
        else
        {
            mstrResponse = "What?";
        }
        bytesSent = Encoding.ASCII.GetBytes(mstrResponse);
        stream.Write(bytesSent, 0, bytesSent.Length);
    }
}
```

سرور زمانی که یک مشتری به آن وصل می‌شود، این کلاس را نمونه‌سازی می‌کند.

۴. یک کلاس به نام `Client` ایجاد کنید که `Form` را پیاده‌سازی می‌کند، و کد زیر را به کلاس بیفزایید:

Visual Basic

کد نمونه:

```
    Shared Sub Connect(ByVal serverIP As String, ByVal message As String)
Dim output As String = ""
Try
            ' Create a TcpClient.
            ' The client requires a TcpServer that is connected
            ' to the same address specified by the server and port
            ' combination.
            Dim port As Int32 = 13
            Dim client As New TcpClient(serverIP, port)

            ' Translate the passed message into ASCII and store it as a byte array.
            Dim data(255) As [Byte]
            data = System.Text.Encoding.ASCII.GetBytes(message)

            ' Get a client stream for reading and writing.
            ' Stream stream = client.GetStream();
            Dim stream As NetworkStream = client.GetStream()

            ' Send the message to the connected TcpServer.
            stream.Write(data, 0, data.Length)

            output = "Sent: " + message
            MessageBox.Show(output)

            ' Buffer to store the response bytes.
            data = New [Byte](255) {}

            ' String to store the response ASCII representation.
            Dim responseData As String = String.Empty

            ' Read the first batch of the TcpServer response bytes.
            Dim bytes As Int32 = stream.Read(data, 0, data.Length)
            responseData = System.Text.Encoding.ASCII.GetString(data, 0,
bytes)
            output = "Received: " + responseData
            MessageBox.Show(output)

            ' Close everything.
            stream.Close()
            client.Close()
        Catch e As ArgumentNullException
            output = "ArgumentNullException: " + e.ToString()
            MessageBox.Show(output)
        Catch e As SocketException
            output = "SocketException: " + e.ToString()
            MessageBox.Show(output)
        End Try

    End Sub

    Shared Sub Main()
        Application.Run(New Client())

    End Sub
```

کد نمونه: **C#**

```
public Client()
        {
            this.MinimizeBox = false;
        }

static void Connect(string serverIP, string message)
        {
            string output = "";

            try
            {
                // Create a TcpClient.
                // The client requires a TcpServer that is connected
                // to the same address specified by the server and port
                // combination.
                Int32 port = 13;
                TcpClient client = new TcpClient(serverIP, port);

                // Translate the passed message into ASCII and store it as a byte
                //array.
                Byte[] data = new Byte[256];
                data = System.Text.Encoding.ASCII.GetBytes(message);

                // Get a client stream for reading and writing.
                // Stream stream = client.GetStream();
                NetworkStream stream = client.GetStream();

                // Send the message to the connected TcpServer.
                stream.Write(data, 0, data.Length);

                output = "Sent: " + message;
                MessageBox.Show(output);

                // Buffer to store the response bytes.
                data = new Byte[256];

                // String to store the response ASCII representation.
                String responseData = String.Empty;

                // Read the first batch of the TcpServer response bytes.
                Int32 bytes = stream.Read(data, 0, data.Length);
                responseData = System.Text.Encoding.ASCII.GetString(data, 0,
bytes);
                output = "Received: " + responseData;
                MessageBox.Show(output);

                // Close everything.
                stream.Close();
                client.Close();
            }
            catch (ArgumentNullException e)
            {
                output = "ArgumentNullException: " + e;
                MessageBox.Show(output);
            }
            catch (SocketException e)
```

```
            {
                output = "SocketException: " + e.ToString();
                MessageBox.Show(output);
            }
        }

static void Main()
        {
            Application.Run(new Client());
        }
```

۵. در کلاس `Client`، روشی را برای کاربر فراهم کنید تا به سرور وصل شود. برای مثال، متد `Connect` را در رویداد Click یک دکمه فراخوان کنید.

Visual Basic کد نمونه:

```
Private Sub button1_Click(ByVal sender As Object, ByVal e As EventArgs)
    ' In this code example, use a hard-coded
    ' IP address and message.
    Dim serverIP As String = "localhost"
    Dim message As String = "Hello"
    Connect(serverIP, message)

End Sub
```

C# کد نمونه:

```
private void button1_Click(object sender, EventArgs e)
{
    // In this code example, use a hard-coded
    // IP address and message.
    string serverIP = "localhost";
    string message = "Hello";
    Connect(serverIP, message);
}
```

۶. برنامه‌های سرور و مشتری را کامپایل کنید.

۷. هر دو برنامه را به دستگاه گسترش دهید (منتقل کنید).

۸. برنامه سرور را روی دستگاه اجرا کرده و سرور را راه‌اندازی کنید.

۹. برنامه مشتری را روی دستگاه اجرا کرده و به سرور متصل شوید.

کامپایل کد

برنامه مشتری (client) نیازمند ارجاعاتی به فضاهای نام زیر است:

- System
- System.Collections.Generic
- System.ComponentModel
- System.Data
- System.Drawing
- System.Text
- System.Windows.Forms
- System.Net.Sockets
- System.Net

برنامه سرور (server) نیازمند ارجاعاتی به فضاهای نام زیر است:

- System
- System.Collections.Generic
- System.ComponentModel
- System.Drawing
- System.Text
- System.Windows.Forms
- System.Net
- System.Net.Sockets
- System.Threading

چگونگی ارسال یک درخواست HTTP با پراکسی

.NET Compact Framework از سرویس‌های وب پشتیبانی می‌کند. این مثال یک درخواست HTTP GET را با یا بدون یک پراکسی مشخص تسلیم می‌کند.

مثال

در کد نمونه زیر، کلیک یک دکمه درخواست را تسلیم کرده و پاسخ را پردازش می‌کند. درخواست تنها در صورتی که پراکسی مشخص شده باشد، از آن استفاده می‌کند و کلاس WebException را برای رسیدگی به هر استثنایی به کار می‌برد. برنامه از یک StreamReader استفاده می‌کند تا پاسخ‌های HTML را به میان بافر آرایه کارکتری بخواند.

کد نمونه: **Visual Basic**

```
Private Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Button1.Click
```

```
    ' Get URL and proxy
    ' from the text boxes.
    Dim url As String = txtURL.Text
    Dim proxy As String = txtProxy.Text

    Try
        If Not "".Equals(txtProxy.Text) Then
            Dim proxyObject As New WebProxy(proxy, 80)

            ' Disable proxy use when the host is local.
            proxyObject.BypassProxyOnLocal = True

            ' HTTP requests use this proxy information.
            GlobalProxySelection.Select = proxyObject
        End If

        Dim req As WebRequest = WebRequest.Create(url)
        Dim result As WebResponse = req.GetResponse()
        Dim ReceiveStream As Stream = result.GetResponseStream()
        Dim encode As Encoding = System.Text.Encoding.GetEncoding("utf-8")
        Dim sr As New StreamReader(ReceiveStream, encode)

        ' Read the stream into arrays of 30 characters
        ' to add as items in the list box. Repeat until
        ' buffer is read.
        Dim read(29) As [Char]
        Dim count As Integer = sr.Read(read, 0, 30)
        While count > 0
            Dim str As New String(read, 0, count)
            lstResults.Items.Add(str)
            count = sr.Read(read, 0, 30)
        End While
    Catch ex As WebException
        Dim message As String = ex.Message
        Dim response As HttpWebResponse = CType(ex.Response, HttpWebResponse)
        If Nothing Is response Then
        Else
            message = response.StatusDescription
            response.Close()
        End If
        lstResults.Items.Add(message)
    Catch ex As Exception
        lstResults.Items.Add(ex.Message)
    End Try
End Sub
```

C#

کد نمونه:

```
private void button1_Click(object sender, System.EventArgs e)
{

    // Get URL and proxy
    // from the text boxes.
    string url = txtURL.Text;
    string proxy = txtProxy.Text;
```

```
    try
    {
        if(!"".Equals(txtProxy.Text))
        {
            WebProxy proxyObject = new WebProxy(proxy, 80);

            // Disable proxy use when the host is local.
            proxyObject.BypassProxyOnLocal = true;

            // HTTP requests use this proxy information.
            GlobalProxySelection.Select = proxyObject;

        }

        WebRequest req = WebRequest.Create(url);
        WebResponse result = req.GetResponse();
        Stream ReceiveStream = result.GetResponseStream();
        Encoding encode = System.Text.Encoding.GetEncoding( "utf-8");
        StreamReader sr = new StreamReader( ReceiveStream, encode );

        // Read the stream into arrays of 30 characters
        // to add as items in the list box. Repeat until
        // buffer is read.
        Char[] read = new Char[30];
        int count = sr.Read( read, 0, 30 );
        while (count > 0)
        {
            String str = new String(read, 0, count);
            lstResults.Items.Add(str);
            count = sr.Read(read, 0, 30);
        }
    }
    catch(WebException ex)
    {
       string message = ex.Message;
       HttpWebResponse response = (HttpWebResponse)ex.Response;
       if(null != response)
       {
           message = response.StatusDescription;
           response.Close();
       }
       lstResults.Items.Add(message);
    }
    catch(Exception ex)
    {
        lstResults.Items.Add(ex.Message);
    }
}
```

کامپایل کد

این مثال نیازمند ارجاعاتی به فضاهای نام زیر است:

- System

- System.Net
- System.Text
- System.Windows.Forms

چگونگی استفاده از یک پراکسی تولید شده توسط Wsdl.exe

.NET Compact Framework از تمامی کدهای تولید شده توسط Web Services Description Language Tool (Wsdl.exe) پشتیبانی نمی‌کند. با این حال، برنامه‌هایی که از سرویس‌های وب استفاده می‌کنند، می‌توانند زمانی که شما در ویژوال استودیو یک ارجاع وب را به یک پروژه Smart Device اضافه می‌کنید، از پراکسی تولید شده استفاده کنند.

زمانی که بخواهید از Wsdl.exe استفاده کنید یک سری وضعیت‌ها می‌تواند پیش بیاید. یک حالت زمانی است که شما بایستی ترتیب اعضای ذره را در پراکسی خود به منظور تطبیق دادن با ترتیب مورد نیاز توسط یک سرویس وب مقرر کنید. ابزار Wsdl.exe یک گزینه **/order** دارد که شناسه‌های ترتیب صریحی را در مورد اعضای ذره تولید می‌کند.

نکته:

به منظور اجتناب از مرحله حذف دستی کد تولید شده از پراکسی، می‌توانید اول یک ارجاع وب (Web reference) را به یک پروژه Smart Device در ویژوال استودیو اضافه کنید، و سپس **Wsdl.exe /order** را در دایرکتوری دیگری اجرا کنید. بالاخره، کدی را که ترتیب صریح عناصر را نگه‌داری کرده است از پراکسی تولید شده توسط Wsdl.exe به میان پراکسی تولید شده توسط اضافه کردن Web reference، کپی کنید.

این مثال نشان می‌دهد که کدام کد باید از پراکسی تولید شده حذف شود به طوری که می‌تواند توسط .NET Compact Framework به کار برده شود. کدی که قرار است حذف شود، به ترتیبی لیست می‌شود که در پراکسی تولید شده ظاهر می‌شود.

برای حذف کد پشتیبانی نشده از پراکسی تولید شده

۱. پراکسی را با Web Services Description Language Tool (Wsdl.exe) تولید کنید.

۲. کدی را که نماینده‌ای را با نام `RetBaseTypesOperationCompleted` از نوع SendOrPostCallback را تعریف می‌کند، حذف کنید.

۳. کدی را که رویداد `RetBaseTypesCompleted`، نماینده `RetBaseTypesCompletedEventHandler`، و کلاس `RetBaseTypesCompletedEventArgs` را تعریف کرده و به آنها ارجاع می‌کند، حذف کنید.

۴. کدی را که متد `RetBaseTypesAsync` را تعریف کرده و فراخوانی می‌کند، حذف کنید.

۵. کدی را که متد `OnRetBaseTypesOperationCompleted` را تعریف کرده و فراخوانی می‌کند، حذف کنید.

۶. کدی را که متد `CancelAsync` را تعریف کرده و فراخوانی می‌کند، حذف کنید.

۷. مشخصه `Serializable` را حذف کنید.

چگونگی استفاده از کنترل WebBrowser در .NET Compact Framework

.NET Compact Framework توابع هسته‌ای مربوط به کنترل WebBrowser برای Windows Forms را پشتیبانی می‌کند. اعضای زیر نیازمند نرم‌افزار Windows Mobile نسخه 5.0 برای Pocket PC یا Smartphone هستند. این لیست در معرض تغییر قرار دارد.

- CanGoBack property
- CanGoForward property
- IsBusy property
- IsOffline property
- ReadyState property
- ScriptErrorsSuppressed property
- CanGoBackChanged event
- CanGoForwardChanged event
- GoBack method
- GoForward method
- OnCanGoBackChanged method
- OnCanGoForwardChanged method
- Refresh method

ملاحظات زیر تنها به Windows Mobile 2003 برای Pocket PC و Windows Mobile 2003 برای Smartphone اعمال می‌شوند:

- متد Refresh یک استثنای NotSupportedException پرتاب می‌کند.

- از آن که کنترل WebBrowser می‌تواند حاوی زیرکنترل‌های تعبیه شده در فرم HTML باشد، شما نمی‌توانید با مانیتورینگ رویداد GetFocus تشخیص دهید که آیا کنترل WebBrowser دارای فوکوس است یا نه. به عنوان یک فعالیت پیرامونی، از فرایند حذف کنترل‌های دیگر که ممکن است دارای فوکوس باشند، استفاده کنید.

- خاصیت **Url** در WebBrowserNavigatingEventArgs تنظیم نمی‌شود و یک رشته تهی را برمی‌گرداند.

روی دستگاهی غیر از یک Pocket PC یا Smartphone که در حال اجرای Microsoft Windows CE 5.0 است، رویدادهای Navigated و DocumentCompleted، زمانی که یک URL تازه ملاقات می‌شود، هر دو اتفاق می‌افتند. پیش بینی شده است که این خطا در ویرایش‌های آتی Windows CE اصلاح شود.

شما می‌توانید نمونه‌ای از WebBrowser را ایجاد کنید تا با پاسخ به رویداد HelpRequested عناوین راهنمای درون خطی را برای برنامه شما نمایش دهد.

.NET Compact Framework. از خاصیت **Document** و خاصیت‌ها و رویدادهای وابسته به آن، به جز خاصیت DocumentText پشتیبانی می‌کند. شما می‌توانید از خاصیت **DocumentText** استفاده کنید تا HTML را برای کاربران خود مهیا کنید، مانند فراهم کردن پیوندها و یک فرم HTML ساده، اما .NET Compact Framework. از دسترسی به محتویات HTML یک صفحه وب با این خاصیت پشتیبانی نمی‌کند.

شما می‌توانید در کدی که رویداد Navigating را اداره می‌کند، پاسخ به فرم را با خاصیت WebBrowserDocumentCompletedEventArgs.Url مشخص کنید. در اولین دستورالعملی که در پی می‌آید، یک کد نمونه تدارک دیده شده است.

شما نمی‌توانید در یک برنامه Smartphone، بیرون از کنترل WebBrowser گشت و گذار (navigate) کنید. به عنوان یک فعالیت پیرامونی، می‌توانید یک رویداد کلیدی (key event) ایجاد کرده و فوکوس را روی کنترل دیگری تنظیم کنید. در دومین دستورالعملی که متعاقباً می‌آید، کد نمونه‌ای مهیا شده است.

نکته:

این فعالیت حاشیه‌ای نیازمند Windows Mobile 5.0 یا 3.5 .NET Compact Framework است.

برای جمع‌آوری اطلاعات از کنترل‌های HTML تعبیه شده

۱. از خاصیت DocumentText استفاده کنید تا HTML را در کنترل WebBrowser نمایش دهید. این HTML حاوی یک فرم همراه با یک پیوند (Link) و یک جعبه متنی به منظور مشخص کردن یک URL است.

Visual Basic کد نمونه:

```
Dim sb As New StringBuilder()
sb.Append("<html><body>")
sb.Append("<a href=")
sb.Append("""")
sb.Append("http://www.microsoft.com")
sb.Append("""")
sb.Append(">Microsoft</a><p>")
sb.Append("Specify a URL:<br>")
sb.Append("<form action=''><input type='text' name='address'/>" )
sb.Append("<br><input type='submit'>")
sb.Append("</form></body></html>")
webBrowser1.DocumentText = sb.ToString()
```

C# کد نمونه:

```
StringBuilder sb = new StringBuilder();
sb.Append("<html><body>");
sb.Append("<a href=");
sb.Append("\"");
sb.Append("http://www.microsoft.com");
sb.Append("\"");
sb.Append(">Microsoft</a><p>");
sb.Append("Specify a URL:<br>");
sb.Append("<form action=''><input type='text' name='address'/>" );
sb.Append("<br><input type='submit'>");
sb.Append("</form></body></html>");
webBrowser1.DocumentText = sb.ToString();
```

۲. از رویداد Navigating استفاده کنید تا مشخص کنید که آیا URL حاوی پاسخی از فرم است. اگر این گونه بود، URL را کاوش نمایید.

Visual Basic کد نمونه:

```
Private Sub webBrowser1_Navigating(ByVal sender As Object, ByVal e As
WebBrowserNavigatingEventArgs)  Handles webBrowser1.Navigating
    Dim x As Integer
```

```
    '  The URL contains the results of the
    '  HTML form following the equals sign.
    x = e.Url.ToString().LastIndexOf("=")
    If x <> - 1 Then
        Dim Redirect As String
        Redirect = e.Url.ToString().Substring((x + 1))
        If Redirect <> "" Then
            ' Error handling code omitted in this example.
            ' Uri constructor throws a UriFormatException if there's
            ' an error.
            webBrowser1.Navigate(New Uri(Redirect))
        Else
            MessageBox.Show("Specify a URL")
        End If
    End If

End Sub
```

C# کد نمونه:

```
private void webBrowser1_Navigating(object sender, WebBrowserNavigatingEventArgs e)
 {
     int x;
     //  The URL contains the results of the
     //  HTML form following the equals sign.
     x = e.Url.ToString().LastIndexOf("=");
     if (x != -1)
     {
         string Redirect;
         Redirect = e.Url.ToString().Substring(x + 1);
         if (Redirect != "")
         {
             // Error handling code omitted in this example.
             // Uri constructor throws a UriFormatException if there's
             // an error.
             webBrowser1.Navigate(new Uri(Redirect));
         }
         else
         {
             MessageBox.Show("Specify a URL");
         }
     }
 }
```

در کد نمونه Visual Basic قبلی، گرداننده رویداد `Navigating` قبلاً با کنترل مرتبط شده است. این گرداننده رویداد را در C# به صورت زیر اعلان کنید:

C# کد نمونه:

```
this.webBrowser1.Navigating += new
System.Windows.Forms.WebBrowserNavigatingEventHandler(this.webBrowser1_Navigating);
```

برای ناوبری (هدایت) به بیرون از WebBrowser روی Smartphone

- کد نمونه زیر، زمانی که UP به اعتبار کلید ناوبری فشرده می‌شود، فوکوس را روی کنترل دیگری تنظیم می‌کند.

کنترل WebBrowsing از منطق حسابرسی از سوی Microsoft Pocket Internet Explorer استفاده می‌کند تا کاربر را قادر سازد تا پیوندهای متفاوت و کنترل‌های تعبیه شده روی وب سایت نشان داده شده توسط کنترل را کاوش کند. شما می‌توانید این رفتار حسابرسی (tabbing) پیش فرض را با استفاده از خاصیت KeyPreview بازتعریف کنید (Override).

در کد نمونه زیر فرض بر این است که در سازنده فرم یا در کدی که رویداد Load مربوط به فرم را اداره می‌کند KeyPreview به **true** تنظیم شده است.

Visual Basic کد نمونه:

```
Protected Overrides Sub OnKeyDown(ByVal keyg As KeyEventArgs)
    If keyg.KeyCode = Keys.Up Then
        textBox1.Focus()
    End If
    MyBase.OnKeyDown(keyg)
```

C# کد نمونه:

```
protected override void OnKeyDown(KeyEventArgs keyg)
{
    if (keyg.KeyCode == Keys.Up)
    {
        textBox1.Focus();
    }
    base.OnKeyDown(keyg);
}
```

کامپایل کد

این مثال نیازمند ارجاعاتی به فضاهای نام زیر است:

- System
- System.Windows.Forms

سازگاری‌های دودویی با .NET Framework کامل

.NET Compact Framework یک پیاده‌سازی زیرمجموعه‌ای سازگار از .NET Framework کامل و CLR است. در غیاب یک سیاست انقیاد strong-name، برنامه‌ای که در مقابل .NET Compact Framework کامپایل می‌شود روی .NET Framework کامل مقید شده و اجرا خواهد شد. به ملاحظات زیر در مورد این دو چارچوب توجه نمایید:

- اسمبلی‌های .NET Compact Framework با جفت کلید Strong-name متفاوت امضا می‌شوند به طوری که CLR می‌تواند آنها را از همتاهای .NET Framework کامل‌شان تمایز دهد.
- .NET Framework سیاست انقیادی عرضه می‌کند تا اسمبلی‌های .NET Framework کامل را به جای ارجاعات .NET Compact Framework سازگار، جایگزین کند. در این صورت در موارد معمولی استفاده از اجزای موجود بدون ایجاد نمونه‌های جدیدی از اشیاء امکان‌پذیر خواهد شد. برای مثال، اگر جزء شما تنها به کلاس‌های .NET Compact Framework System ارجاع کند، با هردو چارچوب سازگار خواهد شد.
- .NET Compact Framework از کلاس‌ها و نوعهایی که تنها برای .NET Compact Framework پشتیبانی می‌شوند و در بخش «کلاس‌های پشتیبانی شده در .NET Compact Framework» لیست شده‌اند، پشتیبانی نمی‌کند.

درست همان طور که سازگاری جفتی پلت‌فرم متقاطع، توسعه و گسترش میان‌افزار را ساده‌سازی می‌کند، برنامه‌های پرمشتری بایستی از کارکردهای مختص دستگاه بهره‌برداری کنند تا توانمندی کاربر را بهبود بخشند. این امر دلالت بر این دارد که بهترین کد رابط کاربری گرافیکی احتمالاً مختص دستگاه هدف خواهد بود.

با این که .NET Compact Framework کارکردهای مختص دستگاهی را میان اسمبلی‌ها و فضاهای نام جدا ازهم و مجرد فاکتورگیری می‌کند تا از تعارضات انقیاد اجتناب کند، احتمالاً مواردی وجود خواهد داشت که

فاکتورگیری ناسازگار نتواند مدیریت شود. در این موارد، استفاده غیرعمدی از کارکردهای مختص دستگاهی با .NET Framework به جای یک استثنای بارگیری برنامه، باعث بروز یک استثنای زمان اجرا خواهد شد.

لیست فایل‌های مربوط به .NET Compact Framework

این بخش اسمبلی‌ها و فایل‌هایی را که توسط .NET Compact Framework برای توسعه برنامه مهیا شده‌اند، لیست می‌کند. این لیست‌ها جامع و فراگیر نبوده و در معرض تغییر قرار دارند.

Common Language Runtime (CLR)

کد خالص

فایل‌های لیست شده در جدول زیر کد خالص بود و بر طبق پردازنده هدف‌گیری شده کامپایل می‌شوند.

فایل	توضیح
Mscoree.dll	ریشه (Stub) مربوط به موتور CLR فعلی.
Mscoree3_5.dll	موتورهای کد و Just-in-time (JIT).
Netcfagl3_5.dll	سهم کد خالص از Windows.Forms.
Netcfd3dm3_5.dll	سهم کد خالص از D3DM.
Cgacutil.exe	برنامه خدماتی نهانگاه اسمبلی سراسری.
نکته:	
Netcf_setup.dll یک فایل CAB می‌باشد، اما روی دستگاه نصب نمی‌شود و بخشی از CLR نمی‌باشد.	

کد مدیریت شده

فایل‌های لیست شده در جدول زیر فایل‌های کد مدیریت شده .NET Framework. کامپایل شده به JIT هستند و مستقل از پردازنده‌اند.

فایل	نسخه	توضیح
Mscorlib.dll	1.0	کتابخانه‌های کلاس مبنا.
System.dll	1.0	کتابخانه‌های کلاس مبنا.
System.xml.dll	1.0	پشتیبانی لازم برای XML Reader، XML Writer و DOM.
System.Core.dll	3.5	کتابخانه‌های کلاس مبنا.
System.Data.dll	1.0	پشتیبانی لازم از مجموعه داده ADO.NET .
System.Drawing.dll	1.0	پشتیبانی لازم برای ترسیم واسط دستگاه گرافیکی (GDI).
System.Messaging.dll	2.0	صف‌بندی پیغام (به عنوان MSMQ نیز شناخته می‌شود).
System.Windows.Forms.dll	1.0	پکیج فرمها.
System.Web.Services.dll	1.0	پشتیبانی لازم از مشتری سرویس وب XML.
Microsoft.VisualBasic.dll	1.0	کتابخانه زمان اجرای Visual Basic.
CustomMarshallers.dll	2.0	جمع کننده‌های سفارشی.
System.ServiceModel.dll	3.5	Windows Communication Foundation (WCF) برای دستگاهها.
System.Xml.Xlinq.dll	3.5	Language-Integrated Query (LINQ) برای دستگاهها.
System.Data.Entity.dll	3.5	از زیرساخت LINQ پشتیبانی می‌کند.
System.Runtime.Serialization.dll	3.5	سریالی و غیرسریالی کردن اشیاء.

کامپوننت‌های اضافی

کد مدیریت شده

فایل‌های لیست شده در جدول زیر، فایل‌های کد مدیریت شده .NET Framework. کامپایل شده به JIT هستند.

این فایل‌ها مستقل از پردازنده بوده و منحصر به .NET Compact Framework. هستند.

فایل	نسخه	توضیح
System.Net.IrDA.dll	1.0	Infrared communications. ارتباطات اینفرارد.
System.Windows.Forms.DataGrid.dll	1.0	**DataGrid** control for the .NET Compact Framework. کنترل **DataGrid** برای .NET .Compact Framework
Microsoft.WindowsCE.Forms.dll	2.0	Windows Embedded CE .NET Framework controls. کنترل‌های Windows Embedded .CE .NET Framework
Microsoft.WindowsMobile.DirectX.dll	2.0	Managed DirectX and Direct3D for devices. مدیریت شده DirectX و Direct3D برای دستگاهها.

Windows Communication Foundation (WCF) Exchange Server mail transport for devices. نقل و انتقال مراسلات Windows Communication Foundation برای (WCF) Exchange Server دستگاهها.	3.5	Microsoft.ServiceModel.Channels.Mail.dll
WCF Exchange Server mail transport for Windows Mobile devices. نقل و انتقال مراسلات WCF برای Exchange Server دستگاههای Windows Mobile.		Microsoft.ServiceModel.Channels.Mail.WindowsMobile.dll

منابع خارجی برای .NET Compact Framework

منابع زیر اطلاعات اضافی را درباره چگونگی توسعه برنامه‌ها با استفاده از .NET Compact Framework در اختیار قرار می‌دهند.

.NET Compact Framework

.NET Compact Framework 3.5 Redistributable چنان چه از قبل در ROM حضور نداشته باشد یا اگر نیازمند به روزرسانی از نسخه 1.0 یا 2.0 به نسخه 3.5 هستید، بایستی روی دستگاه موبایل شما نصب گردد. این نرم‌افزار دربردارنده تمام اجزایی است که شما به آنها نیاز دارید تا برنامه‌های .NET Compact Framework نسخه‌های 1.0، 2.0 و 3.5 را اجرا کنید. برای دانلود این نرم‌افزار پیوند « .NET Compact Framework Downloads» را در مرکز دانلود مایکروسافت ملاقات نمایید.

مرکز توسعه موبایل

برای آگاهی از تازه‌ترین اطلاعات درباره توسعه برنامه برای موبایل و دستگاه‌های جاسازی شده، پیوند «Microsoft .NET Framework Developer Center» را در سایت www.Microsoft.com ملاقات کنید. صفحه .NET Compact Framework را برای دانلود و سرویس پک‌ها، اطلاعاتی درباره محصول، تجدید نظرها، مقاله‌های تکنیکی، گروه های خبری اجتماعی و دیگر منابع ملاحظه کنید.

هم چنین چیزهای دیگری که در دسترس می‌باشند، اطلاعاتی درباره SDKهای Windows Mobile دانلودهاست. صفحات مربوط به Windows Mobile و .NET Compact Framework می‌توانند از قاب ناوبری **Key Technologies** واقع در Mobile Developer Center، قابل دسترسی باشند.

SDKهای Windows Mobile

برای ایجاد برنامه‌ها با استفاده از برنامه‌نویسی Mobile Direct3D مدیریت شده، به Windows Mobile 5.0 SDK برای Pocket PC و Windows Mobile 5.0 SDK برای Smartphone نیاز خواهید داشت. ضمناً برای توسعه برنامه‌ها با استفاده از .NET Compact Framework، نسخه 3.5، نیازمند Windows Mobile 5.0 SDK برای Smartphone هستید.

SDKها را همراه با Microsoft Visual Studio 2008 روی یک کامپیوتر نصب کنید. زمانی که پروژه جدیدی را ایجاد می‌کنید، می‌توانید یکی از انواع پروژه **Windows Mobile 5.0 for Pocket PC** و **Windows Mobile 5.0 for Smartphone** را انتخاب کنید و ضمناً برنامه را به نمونه‌سازهای Windows Mobile 5.0 گسترش دهید.

کتابخانه کلاس Microsoft.WindowsMobile مجموعه‌ای از کلاس‌ها، برشمارش‌ها (ریزفهرست‌ها) و نماینده‌هاست که بخش مدیریت شده Microsoft Windows Mobile SDK را شکل می‌دهند. فضاهای نام زیر به عنوان بخشی از Windows Mobile 5.0 SDK تنها برای Pocket PC و Smartphoneها در دسترس هستند.

```
Microsoft.WindowsMobile.Configuration
```

کلاسی را برای آزمایش طراحی فایل‌های پیکربندی XML که به منظور تدارک دستگاه‌های موبایل به کار برده می‌شوند، ارائه می‌دهد.

Microsoft.WindowsMobile.Forms

کلاس‌هایی را برای ایجاد جعبه‌های محاوره‌ای سفارشی به منظور انتخاب اتصالات و گزینش تصاویر عرضه می‌کند.

Microsoft.WindowsMobile.PocketOutlook

کلاس‌هایی را عرضه می‌کند که شما را قادر می‌سازد تا قرارهای ملاقات، وظایف، ارتباطات، پیغام‌های MAPI و پیغام‌های SMS را روی Pocket PCها و Smartphoneها ایجاد کرده و به آنها دسترسی پیدا کنید.

Microsoft.WindowsMobile.PocketOutlook.MessageInterception

کلاس‌هایی را عرضه می‌کند که دسترسی لازم به یک مکانیزم پردازش پیغام برای رهگیری خودکار پیغام‌های SMS ورودی با خاصیت‌هایی که معیارهای خاصی را مطابقت می‌دهند، و پس از آن برنامه‌ها شروع به پردازش آنها می‌کنند، در اختیار شما قرار می‌دهد.

Microsoft.WindowsMobile.Status

به منظور دسترسی و نگه‌داری خاصیت‌های سیستم و نیز تعریف فعالیت‌های برنامه‌ای برای تفسیر بر طبق معیارهایی که تعریف می‌کنید، کلاس‌ها، نماینده‌ها و ریزفهرست‌هایی را عرضه می‌کند.

Microsoft.WindowsMobile.Telephony

کلاسی را تعریف می‌کند که دارای متدی برای گماردن یک مکالمه تلفنی است.

چگونگی مشخص کردن اعضای پشتیبانی شده در کتابخانه کلاس

شما می‌توانید متدهای زیر را برای کسب اطلاعات کتابخانه کلاسی که مختص .NET Compact Framework هستند، به کار برید:

- از فیلترهای ارائه شده در ناظران Help محلی استفاده کنید.
- به اطلاعات پلت‌فرم و نسخه ضمیمه شده با مستندات مربوط به انواع و اعضای اختصاصی مراجعه کنید.

فیلترهای ناظر Help محلی شما را قادر می‌سازند تا به راحتی تشخیص دهید که کدام فضاهای نام، انواع و اعضا توسط .NET Compact Framework پشتیبانی می‌شوند. توجه کنید که این فیلترها تنها در ناظر Help محلی Windows Software Development Kit (SDK) یا مجموعه Help ترکیبی ویژوال استودیو در دسترس می‌باشند.

علاوه بر فیلترهای ناظر Help محلی، بخش‌های متعددی در عناوین نوع و عضو، اطلاعاتی را در مورد پشتیبانی پلت‌فرم دستگاه، پشتیبانی نسخه پلت‌فرم، و در برخی از موارد، رفتارهای مختص پلت‌فرم در اختیار قرار می‌دهند:

- در انتهای بخش "Remarks"، شما می‌توانید هر گونه نکته پلت‌فرمی را که می‌تواند اعمال شود، مانند این چگونه یک متد به خصوص روی یک پلت‌فرم دستگاه موبایل عمل می‌کند، پیدا کنید.
- بخش "Platforms"، Windows Mobile برای Pocket PC، Windows Mobile برای Smartphone و Windows Embedded CE برای نوعها و اعضایی را که توسط .NET Compact Framework پشتیبانی می‌شوند، لیست می‌کند. Windows Embedded CE حاوی دستگاههایی است که Windows Embedded CE را اجرا می‌کنند اما Pocket PC یا Smartphone نیستند.
- بخش "Version Information" نشان می‌دهد که کدام نسخه از .NET Compact Framework. آن نوع یا عضو را پشتیبانی می‌کند.

نکته:

برای تعداد کمی از اعضا، بخش "Version Information" در یک بخش ارجاعی پشتیبانی از .NET Compact Framework را مشخص می‌کند، اما برخی یا همه پلت‌فرم‌های دستگاه موبایل در بخش "Platforms" غایب هستند. این امر نشان می‌دهد که عضو در اسمبلی‌های .NET Compact Framework حضور دارد، اما تنها روی زیرمجموعه‌ای از پلت‌فرم‌ها (صفر یا بیشتر) پشتیبانی می‌شود.

برای مشاهده لیستی از همه پلت‌فرمهای دستگاه و نسخه‌های پلت‌فرم پشتیبانی شده توسط .NET Compact Framework، بخش «دستگاهها و پلت‌فرم‌های پشتیبانی شده توسط .NET Compact Framework» را ملاحظه نمایید.

فیلترهای .NET Compact Framework و Smart Device Development

فیلتری که می‌توانید استفاده کنید تا نشان دهید که کدام فضاهای نام، نوعها و اعضا توسط .NET Compact Framework پشتیبانی می‌شوند بستگی به محصولی دارد که در حال استفاده از آن هستید.

- در Windows SDK، از فیلتر .NET Compact Framework استفاده کنید.
- در Visual Studio، از فیلتر Smart Device Development استفاده کنید.

برای استفاده از فیلتر .NET Compact Framework یا Smart Device Development

- در قاب **Index** یا **Contents** در Help، در لیست **Filter by**، مورد **.NET Compact Framework** یا **Smart Device Development** را کلیک کنید.

این دو فیلتر اطلاعاتی را که در جدول contents و index نمایش داده می‌شود، محدود می‌کند. آنها صرفاً اطلاعات زیر را نمایش می‌دهند:

- تنها فضاهای نامی که حاوی انواع پشتیبانی شده توسط .NET Compact Framework هستند.
- تنها انواع واقع در یک فضای نام که توسط .NET Compact Framework پشتیبانی می‌شوند.
- تنها سربارگذاری‌های متد و سازنده‌ای که توسط .NET Compact Framework پشتیبانی می‌شوند.

نکته:

علیرغم این که فیلترینگ به کار برده شده است، اعضای به ارث رسیده در جدول contents ظاهر نمی‌شوند.

- تنها محتویات مفهومی که مربوط به .NET Compact Framework هستند.

چگونگی تحصیل دایرکتوری برنامه

از آن جایی که هیچ گونه تنظیمات دایرکتوری جاری ذاتی در برنامه‌های Pocket PC وجود ندارد، مشخص کردن یک نام فایل در کد بدون هیچ مشخصات مسیری یک استثنای FileNotFoundException را برمی‌گرداند. برنامه‌های Pocket PC فایل‌های داده را همراه با فایل‌های اسمبلی شما تحت \Program Files*myAssembly*\ ذخیره می‌کنند، جایی که *myAssembly* نام اسمبلی شماست.

مثال

این مثال چگونگی مشخص کردن مسیر برنامه در حال اجرای فعلی را با استفاده از نام دایرکتوری کاملاً توصیف شده از اسمبلی اجرایی، نمایش می‌دهد. توجه کنید که چنان چه برنامه در حال اجرا شدن در دایرکتوری ریشه دستگاه باشد، اطلاعات مسیر برگشتی یک رشته تهی خواهد بود.

Visual Basic کد نمونه:

```
Dim strAppDir As String = _
  Path.GetDirectoryName([Assembly].GetExecutingAssembly().GetModules(0).FullyQualifiedName)
MsgBox(strAppDir)
```

C# کد نمونه:

```
string strAppDir =
Path.GetDirectoryName(Assembly.GetExecutingAssembly().GetModules()[0].FullyQualifiedName);
MessageBox.Show(strAppDir);
```

کامپایل کد

این مثال نیازمند ارجاعاتی به فضاهای نام زیر است:

- System
- System.Windows.Forms
- System.IO
- System.Reflection

چگونگی اداره تغییرات جهت‌گیری و تفکیک‌پذیری

جدول زیر جهت‌گیری‌های صفحه نمایش Pocket PC و Smartphone پشتیبانی شده توسط .NET Compact Framework را نشان می‌دهد. ابعاد در واحد پیکسل و به صورت عرض در ارتفاع هستند.

جهت‌گیری صفحه نمایش	ابعاد نوعی با **96 DPI**	ابعاد نوعی با **192 DPI** (تفکیک‌پذیری بالا)
Portrait - Pocket PC Portrait - Smartphone	240 x 320 176 x 220, and 240 x 320	480 x 640 352 x 440, and 480 x 640
Landscape - Pocket PC only	320 x 240	640 x 480
Square - Pocket PC only	240 x 240	480 x 480

شما می‌توانید از Anchoring (مهارسازی) و Docking (موقوف‌سازی) استفاده کنید تا به طور خودکار کنترل‌ها را تغییر اندازه دهید طوری که محتویات فرم خود را با جهت‌گیری‌های صفحه نمایش متفاوت وفق دهند. Anchoring فاصله مشخصی را از یک لبه حفظ می‌کند، در حالی که Docking به لبه یک کانتینر والد می‌چسبد.

خاصیت **Anchor** یک کنترل زمانی که کنترل یا فرم متضمن و دربردارنده‌اش تغییر اندازه داده می‌شود، رفتا تغییر اندازه‌اش را تعیین می‌کند. خاصیت **Dock** یک کنترل لبه‌هایی از کنترل دربردارنده‌اش را مشخص می‌کند که کنترل باید به آن بچسبد.

مهارسازی (Anchoring) و موقوف‌سازی (Docking) در .NET Compact Framework. همان رفتاری را دارند که .NET Framework. کامل دارند. شما می‌توانید برنامه خود را سفارشی کنید تا خود را با تنظیمات تفکیک‌پذیری پیکسل DPI (نقطه به ازای اینچ) متفاوت وفق دهد.

به منظور اداره یک تغییر در جهت‌گیری صفحه نمایش

- شما می‌توانید کنترل‌هایی را که باید در ناحیه به خصوصی از فرم باشند در یک کنترل کانتینر، مانند یک Panel جای دهید و سپس خاصیت Dock کنترل Panel را به لبه دلخواه تنظیم کنید.
- برای نگه‌داری اندازه و موقعیت درست یک کنترل نسبت به کناره‌های فرمش، خاصیت Anchor را روی کنترل‌ها به موقعیت مطلوب تنظیم کنید.

برای مثال، برای این دکمه‌ای داشته باشید که همواره در گوشه سمت راست پایینی ظاهر شده و اندازه‌اش را حفظ کند، از دستور زیر استفاده کنید:

Visual Basic کد نمونه:

```
Me.Button1.Anchor = AnchorStyles.Bottom Or AnchorStyles.Right
```

C# کد نمونه:

```
this.button1.Anchor = AnchorStyles.Bottom | AnchorStyles.Right;
```

در Microsoft Visual Studio 2008، شما می‌توانید تنظیمات مهارسازی و موقوف‌سازی را در قاب **Properties** انجام دهید.

برای تغییر جهت‌گیری صفحه نمایش

- چنان چه Pocket PC شما در حال اجرای Windows Moblie نسخه 5.0 باشد، می‌توانید جهت‌گیری صفحه نمایش را از Portrait در صفر درجه تا ۹۰، ۱۸۰ و ۲۷۰ درجه به صورتی که توسط ریزفهرست ScreenOrientation مشخص شده، تغییر دهید. برای مثال، دستور زیر یک جهت‌گیری Landscape را تنظیم می‌کند:

Visual Basic کد نمونه:

```
SystemSettings.ScreenOrientation = ScreenOrientation.Angle270
```

C# کد نمونه:

```
SystemSettings.ScreenOrientation = ScreenOrientation.Angle270;
```

توجه کنید که جهت‌گیری صفحه نمایش دستگاه را تغییر می‌دهد، نه فقط برنامه را. از این رو، یک تمرین خوب می‌تواند این باشد که در کد رسیدگی به رویداد مربوط به رویداد FormClosing، جهت‌گیری صفحه نمایش را به تنظیمات اصلی‌اش برگردانید.

اداره یک تغییر در تفکیک‌پذیری صفحه نمایش

- زمانی که یک پروژه Smart Device را در Microsoft Visual Studio 2008 ایجاد می‌کنید، طراح کدی را در اختیار قرار می‌دهد که به طور خودکار کنترل‌ها را به طرز مناسبی برای تفکیک‌پذیری صفحه نمایش دستگاه مقیاس‌بندی می‌کند؛ در غیر این صورت، چنان چه برنامه شما در حال اجرا روی یک دستگاه با یک تفکیک‌پذیری DPI متفاوتی از دستگاه به کار رفته برای توسعه آن باشد، فرم یا خیلی بزرگ یا خیلی کوچک ظاهر خواهد شد. در نتیجه، کنترل‌های واقع در روی فرم بایستی به طور مقتضی مقیاس‌بندی شوند.

زمانی که یک پروژه Smart Device را در Microsoft Visual Studio 2008 ایجاد می‌کنید، طراح دستورات زیرا به متد `InitializeComponent` اضافه می‌کند:

[Visual Basic]

کد نمونه:

```
Me.AutoScaleDimensions = New System.Drawing.SizeF(96.0!, 96.0!)
Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Dpi
```

[C#]

کد نمونه:

```
this.AutoScaleDimensions = new System.Drawing.SizeF(96F, 96F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Dpi;
```

این دستورات برنامه‌ای را نشان می‌دهد که برای دستگاهی با یک تفکیک‌پذیری 96 DPI طراحی شده و این که مقیاس‌بندی اتوماتیک با استفاده از حالت DPI انجام می‌شود. توجه کنید که 96 DPI مقدار پیش فرض به دست آمده از کامپیوتر رومیزی در حال اجرای Visual Studio 2008 است. طراح کد به طور خودکار این کد را تولید می‌کند، و کنترل‌های واقع در روی فرم به طور خودکار مقیاس‌بندی می‌شوند تا تنظیمات DPI متفاوتی را مدیریت کنند.

چنان چه برنامه شما حاوی گرافیک‌هایی است که در متد OnPaint ترسیم می‌شوند، این گرافیک‌ها به طور خودکار مقیاس‌بندی نخواهند شد. شما نیاز خواهید داشت که از خاصیت‌های **DpiX** و **DpiY** اشیاء Graphics خود استفاده کنید تا مقیاس‌بندی درخور و شایسته‌ای را مشخص کنید.

ریزفهرست ScreenOrientation

زاویه جهت‌گیری صفحه نمایش دستگاه را تعیین می‌کند که می‌تواند با خاصیت ScreenOrientation مورد دسترسی واقع شود.

Namespace: Microsoft.WindowsCE.Forms
Assembly: Microsoft.WindowsCE.Forms (in Microsoft.WindowsCE.Forms.dll)

اعضای ریز فهرست ScreenOrientation

	نام عضو	توضیح
	Angle0	یک جهت‌گیری Portrait را در صفر درجه مشخص می‌کند.
	Angle90	یک جهت‌گیری را در ۹۰ درجه مشخص می‌کند.
	Angle180	یک جهت‌گیری Landscape را در ۱۸۰ درجه مشخص می‌کند.
	Angle270	یک جهت‌گیری را در ۲۷۰ درجه مشخص می‌کند.

نکات

جهت‌گیری Portrait پیش فرض در زاویه صفر درجه است. Pocket PC 2003 نیازمند به روزرسانی است.

مثال‌ها

کد نمونه زیر نشان می‌دهد که چگونه با کلیک دکمه‌ای می‌توان جهت‌گیری صفحه نمایش را از طریق ریزفهرست **ScreenOrientation** تغییر داد.

Visual Basic کد نمونه:

```
' Each click event changes the screen orientation, as determined
' by the variable x, which increments from 0 to 3 and then back
' to 0. Four clicks cycle through the ScreenOrientation enumeration.
Private Sub Button1_Click(sender As Object, e As System.EventArgs) Handles
Button1.Click

   Select Case x
      Case 0
```

```
            ' Pass a value for the ScreenOrientation enumeration
            ' to the SetOrientation method, defined below,
            ' and increment x so that the next button
            ' click rotates the screen orientation.
            SetOrientation(ScreenOrientation.Angle90)
            x += 1
         Case 1
            SetOrientation(ScreenOrientation.Angle180)
            x += 1
         Case 2
            SetOrientation(ScreenOrientation.Angle270)
            x += 1
         Case 3
            SetOrientation(ScreenOrientation.Angle0)
            x = 0
         Case Else
            SetOrientation(ScreenOrientation.Angle0)
            x = 0
   End Select
End Sub


' Set the orientation to a value of the
' ScreenOrienation enumeration and update the
' status bar with the current angle.
Private Sub SetOrientation(so As ScreenOrientation)
   ' Set the requested orientation.
   SystemSettings.ScreenOrientation = so

   Me.StatusBar1.Text = SystemSettings.ScreenOrientation.ToString()
End Sub
```

کد نمونه: **C#**

```
 // Each click event changes the screen orientation, as determined
 // by the variable x, which increments from 0 to 3 and then back
 // to 0. Four clicks cycle through the ScreenOrientation enumeration.

private void button1_Click(object sender, System.EventArgs e)
{

 switch(x)
 {
  case 0:

   // Pass a value for the ScreenOrientation enumeration
   // to the SetOrientation method, defined below,
   // and increment x so that the next button
   // click rotates the screen orientation.
   SetOrientation(ScreenOrientation.Angle90);
   x++;
   break;
  case 1:
   SetOrientation(ScreenOrientation.Angle180);
   x++;
   break;
  case 2:
```

```
    SetOrientation(ScreenOrientation.Angle270);
    x++;
    break;
   case 3:
    SetOrientation(ScreenOrientation.Angle0);
    x = 0;
    break;
   default:
    SetOrientation(ScreenOrientation.Angle0);
    x = 0;
    break;
  }
}

 // Set the orientation to a value of the
 // ScreenOrienation enumeration and update the
 // status bar with the current angle.
private void SetOrientation(ScreenOrientation so)
{
 // Set the requested orientation.

 SystemSettings.ScreenOrientation = so;

 this.statusBar1.Text = SystemSettings.ScreenOrientation.ToString();
}
```

پلت‌فرم‌هایی که از این ریزفهرست پشتیبانی می‌کنند.

Windows CE، Windows Mobile برای Smartphone، Windows Mobile برای Pocket PC.

.NET Framework و .NET Compact Framework از تمامی نسخه‌های هر پلت‌فرم پشتیبای نمی‌کنند.

اطلاعات نسخه

.NET Compact Framework

توسط نسخه‌های 2.0 و 3.5 پشتیبانی می‌شود.

صف‌بندی پیغام در .NET Compact Framework

شما می‌توانید از مؤلفه Windows CE MSMQ در .NET Compact Framework استفاده کنید، و ضمناً می‌توانید ویژگی‌های ضمیمه شده در نوعهای واقع در فضای نام System.Messaging را به کار برید.

فضای نام **System.Messaging** کلاس‌هایی را عرضه می‌کند که به شما اجازه می‌دهند تا روی شبکه به صف‌های پیغام وصل شده، آنها را مانیتور کرده و اداره‌شان نمایید و پیغام‌ها را ارسال کرده، دریافت نمایید یا (به صورت دزدکی) تحت نظرشان بگیرید.

MSMQ در .NET Compact Framework

.NET Compact Framework نسخه 2.0 از صف‌بندی پیغام (که به عنوان MSMQ نیز شناخته می‌شود) در Windows CE پشتیبانی می‌کند. MSQM برنامه‌ها را قادر می‌سازد تا با برنامه‌های دیگر در عرض شبکه‌ها و سیستم‌هایی که ممکن است موقتاً آفلاین باشند، ارتباط برقرار کنند.

برای استفاده از این سرویس، برنامه‌ها پیغام‌ها را به یک صف پیغام ارسال می‌کنند. صف پیغام می‌تواند پیغامهایی را هم برای برنامه‌های ارسال کننده و هم دریافت کننده نگه دارد و این برنامه‌ها یا روی دستگاه یکسانی قرار دارند یا روی دستگاههای متفاوتی واقع شده‌اند.

زمانی که یک اتصال شبکه برقرار می‌شود، MSMQ پیغام‌ها را علیرغم این که یک برنامه دریافت کننده درحال اجرا باشد یا نه، پیغام‌ها را به صف دوردست (remote queue) تحویل می‌دهد. برنامه دریافت کننده می‌تواند در هر زمانی در صف محلی خودش، پیغام‌ها را بررسی کند.

MSMQ در میان سیستم عامل Windows CE روی یک Pocket PC تعبیه نشده است، اما شما می‌توانید آن را نصب کنید، سرویس را راه‌اندازی کنید و برنامه‌هایی را که از آن استفاده می‌کنند، ایجاد کنید. رویه زیر چگونگی به دست آوردن مؤلفه MSMQ را تشریح می‌کند:

- Windows Mobile 2003 برای Pocket PC SDK، جزء MSMQ را Pocket PCهای در حال اجرای Windows Mobile عرضه می‌کند.
- دستگاههای در حال اجرای نرم‌افزار Windows Mobile Version 5.0 برای Pocket PCها و Smartphoneها، می‌توانند جزء MSMQ را از «http://go.microsoft.com/fwlink/?LinkId=53900» دانلود کنند.

- MSMQ مربوط به Windows Mobile برای Smartphone در دسترس نیست.
- MSMQ با استفاده از Microsoft Platform Builder برای توسعه‌دهندگان Windows CE در دسترس است.

Windows Mobile 2003 برای Pocket PC از پروتکل SRMP (پروتکل ارسال پیغام قابل اعتماد SOAP) مبتنی بر HTTP پشتیبانی نمی‌کند، پس شما بایستی از یک پروتکل MSMQ اختصاصی برای ارسال پیغام‌ها استفاده کنید. SRMP توسط Windows Mobile نسخه 5.0 پشتیبانی می‌شود.

.NET Compact Framework. تنها XmlMessageFormatter را برای مرتب کردن و نامرتب کردن پیغام‌ها به/از صف پیغام به کار می‌برد.

.NET Compact Framework. از ویژگی‌های زیر پشتیبانی نمی‌کند، زیرا این مشخصه‌ها در Windows CE قابل دسترسی نیستند:

- تبادلات پیغام چندگانه. پشتیبانی از تبادل، به تبادل پیغام واحد محدود می‌شود.
- خواندن صف از راه دور.
- حفاظت یا رمزداز کردن (Encryption).
- امنیتی مبتنی بر یک Access Control List (ACL).
- MQMail.
- صف‌های عمومی مبتنی بر Active Directory.

از آن جایی که Active Directory روی دستگاه‌ها پشتیبانی نمی‌شود، .NET Compact Framework. نمی‌تواند تشخیص دهد که آیا یک صف دورست (remote queue) تبادلی است یا نه. برای ارسال یک صف تبادلی دوردست، دستورالعمل زیر مورد نیاز است:

۱. **XACTONLY;** را به مسیر صف در سازنده‌های MessageQueue که یک پارامتر مسر رشته‌ای را دریافت می‌کند، اضافه کنید و آن را به خاصیت Path بیفزایید.

۲. Single را برای MessageQueueTransactionType در متدهای Send که آن پارامتر را می‌گیرند، مشخص کنید.

.NET Compact Framework از انواع زیر در فضای نام System.Messaging پشتیبانی نمی‌کند:

AccessControlEntry	MessageQueueEnumerator
AccessControlEntryType	MessageQueueInstaller
AccessControlList	MessageQueuePermission
ActiveXMessageFormatter	MessageQueuePermissionAttribute
BinaryMessageFormatter	MessageQueuePermissionEntry
CryptographicProviderType	MessageQueuePermissionEntryCollection
Cursor	MessageQueueTransaction
EncryptionAlgorithm	MessageQueueTransactionStatus
EncryptionRequired	MessagingDescriptionAttribute
GenericAccessRights	PeekAction
HashAlgorithm	QueueAccessMode
MessageLookupAction	StandardAccessRights
MessageQueueAccessControlEntry	Trustee
MessageQueueAccessRights	TrusteeType
MessageQueueCriteria	

چگونگی استفاده از MSMQ در .NET Compact Framework

ایجاد برنامه‌های .NET Compact Framework که از صف‌بندی پیغام (که به عنوان MSMQ نیز شناخته می‌شود) استفاده می‌کنند، مشابه فرایندی است که در .NET Framework به کار برده می‌شود. گرچه، Windows CE از تمامی مشخصه‌های تشریح شده در بخش «MSMQ در .NET Compact Framework» پشتیبانی نمی‌کند.

The following code examples show how to create a message queue, send a message to the queue, and receive a message from the queue, all on the same device. No network connectivity is required. A simple class, `Order`, is used to create objects for Message Queuing.

نمونه کدهای زیر چگونگی ایجاد یک صف پیغام، ارسال یک پیغام به صف، و دریافت یک پیغام از صف، همگی روی دستگاهی یکسان را نشان می‌دهد. هیچ ارتباط شبکه‌ای مورد نیاز نیست. یک کلاس ساده، به نام `Order`، برای ایجاد اشیائی برای صف‌بندی پیغام به کار برده می‌شود.

در این مثال‌ها فرض بر این است که Message Queuing روی دستگاه نصب شده است. برای آگاهی از اطلاعات بیشتر درباره کسب مؤلفه Message Queuing، بخش «MSMQ در .NET Compact Framework» را ملاحظه نمایید.

تعریف کلاس Order

- کلاس زیر را به پروژه خود اضافه کنید.

Visual Basic کد نمونه:

```
' This class represents an object that
' is sent to and received from the queue.
Public Class Order
    Dim ID As Integer
    Dim DTime As DateTime
    Public Property orderID() As Integer
        Get
            Return Me.ID
        End Get
        Set(ByVal value As Integer)
            Me.ID = value
        End Set
    End Property
    Public Property orderTime() As DateTime
        Get
            Return Me.DTime
        End Get
        Set(ByVal value As DateTime)
            Me.DTime = value
        End Set
    End Property
End Class
```

ایجاد صف پیغام

- متد زیر را به فرم خود بیفزایید.

Visual Basic کد نمونه:

```
Private Sub CreateQueue()
    ' Determine whether the queue exists.
    If Not MessageQueue.Exists(QPath) Then
        Try
            ' Create the queue if it does not exist.
            myQ = MessageQueue.Create(QPath)
            MsgBox("Queue Created")
        Catch ex As Exception
```

```
            MsgBox(ex.Message)
        End Try
    Else
        MsgBox("Queue Exists")
    End If
End Sub
```

برای ارسال یک پیغام به صف

- متد زیر را به فرم خود اضافه کنید.

Visual Basic کد نمونه:

```
Private Sub SendMessageToQueue()
    ' Create a new order and set values.
    Dim sendOrder As New Order()
    sendOrder.orderID = 23123
    sendOrder.orderTime = DateTime.Now
    Try
        myQ.Send(sendOrder)
        MsgBox("Message Sent")
    Catch ex As Exception
        MsgBox(ex.Message)
    End Try
End Sub
```

برای دریافت پیغام از صف

- متد زیر را به فرم خود اضافه نمایید.

Visual Basic کد نمونه:

```
Private Sub ReceiveMessageFromQueue()
    ' Connect to the a queue on the device.
    myQ = New MessageQueue(QPath)

    ' Set the formatter to indicate the body contains an Order.
    Dim targetTypes() As Type
    targetTypes = New Type() {GetType(Order)}
    myQ.Formatter = New XmlMessageFormatter(targetTypes)
    Try
        ' Receive and format the message.
        Dim myMessage As Message = myQ.Receive()
        Dim myOrder As Order = CType(myMessage.Body, Order)

        ' Display message information.
        MsgBox("Order ID: " & _
            myOrder.orderID.ToString() & _
                Chr(10) & "Sent: " & myOrder.orderTime.ToString())
```

```
    Catch m As MessageQueueException
        ' Handle Message Queuing exceptions.
        MsgBox(m.Message)
    Catch e As InvalidOperationException
        ' Handle invalid serialization format.
        MsgBox(e.Message)
    End Try
End Sub
```

برای آزمایش صف‌بندی پیغام

۱. دکمه‌ای را که با **Send** برچسب‌دار شده است، به فرم اضافه کنید، تا متدهای `CreateQueue` و `SendMessageToQueue` را فراخوانی کند.

۲. دکمه‌ای را که با **Receive** برچسب‌دار شده است، به فرم اضافه کنید، تا متد `ReceiveMessageFromQueue` را فراخوانی کند.

کامپایل کد

این مثال نیازمند ارجاعاتی به فضاهای نام زیر است:

- System
- System.Messaging
- System.Windows.Forms

مدیریت ریسمان در .NET Compact Framework

.NET Compact Framework از عمل ریسمان‌سازی (Threading) مرکزی پشتیبانی می‌کند اما از ویژگی‌های زیر در .NET Framework کامل پشتیبانی نمی‌کند:

- دسترسی به پشته فشرده بر روی ریسمان جاری.
- نمایش مدیریت شده‌ای از ساختار **OVERLAPPED** برای Win32.
- کلاس‌هایی که بافت اجرا را مدیریت می‌کنند.
- استفاده از سمافورها (Semaphores).
- صفات حالت ریسمان.

- اغلب عملیات‌های دستگیره انتظار به جز برای گرفتن دستگیره.

NET Compact Framework. از به کار بردن نماینده ThreadStart برای استفاده با متد Thread.Start و استفاده از نماینده TimerCallback با یک Timer پشتیبانی می‌کند.

NET Compact Framework. نسخه 2.0 از مشخص کردن پارامترهای وقفه زمانی (Time-out) برای متدهای ریسمان‌سازی زیر پشتیبانی می‌کند:

- متد WaitOne(Int32, Boolean) از یک کلاس WaitHandle.
- متد Join(Int32) از یک کلاس Tread.

تغییرات در مخازن ریسمان

در NET Compact Framework. نسخه 1.0، تعداد حداکثری پیش فرض ریسمان‌ها در یک مخزن ریسمان برابر با ۲۵۶ با یک اندازه پشته 64K است. در نسخه 2.0، این مقدار تا ۲۵ با یک اندازه پشته 128K کاهش می‌یابد، که با دقت بیشتری با عملکرد NET Framework. (رومیزی) کامل مطابقت دارد.

یک درخواست HTTP چنان چه هیچ گونه ریسمانی در مخزن ریسمان در دسترس نباشد، نافرجام مانده و یک استثناء پرتاب خواهد کرد. استثناهای اصلی در این وضعیت موارد زیرند.

استثناء	شرایط بروز
OutOfMemoryException	قادر به صف‌بندی مراجعه مجدد کاربر نباشد.
WebException	برای کامل کردن عملیات، در مخزن ریسمان، ریسمان کافی وجود نداشته باشد.

با کاهش تعداد درخواست‌های وب همزمان، یا با افزایش حداکثر ریسمان‌های مجاز واقع در مخزن ریسمان، می‌توانید از کم آوردن (یا تمام کردن) ریسمان‌ها اجتناب کنید. NET Compact Framework. نسخه 2.0 از متد SetMaxThreads پشتیبانی می‌کند. پارامترهای این متد را به این صورت مشخص نمایید:

پارامتر	توضیح
workerThreads	حداکثر تعداد ریسمان‌های عمل کننده در مخزن ریسمان. این پارامتر می‌تواند هر مقداری باشد.
completionPortThreads	حداکثر تعداد ریسمان‌های غیرهمزمان در مخزن ریسمان. .NET Compact Framework فعلاً از این مقدار چشم‌پوشی می‌کند، اما با مقداری مابین ۱ و ۱۰۰۰ تنظیم شود. برای سازگاری با آینده، مقدار ۵۰۰ توصیه می‌شود زیرا این عدد مقدار پیش فرض .NET Framework کامل است.

برای دستگاه‌هایی که در حال اجرای .NET Compact Framework نسخه 1.0 هستند، شما می‌توانید با تغییر یک تنظیم رجیستری، حداکثر ریسمان‌های مجاز را در مخزن ریسمان کاهش دهید. MaxThread را در کلید **CFROOT\ThreadPool** به مقدار مطلوب تنظیم کنید. توجه کنید که این کلید رجیستری توسط .NET Compact Framework نسخه 2.0 به کار برده نمی‌شود.

فضای نام System.Threading

فضای نام **System.Threading** کلاس‌ها و واسط‌هایی را عرضه می‌کند که برنامه‌نویسی چند ریسمانی را امکان‌پذیر می‌کنند. علاوه بر کلاس‌های مربوط به همزمان‌سازی فعالیت‌های ریسمان و دسترسی به داده (Mutex، Monitor، Interlocked، AutoResetEvent و ...)، این فضای نام حاوی کلاس ThreadPool است که به شما اجازه استفاده از مخزنی از ریسمان‌های عرضه شده با سیستم را می دهد، و یک کلاس Timer که متدهای مراجعه مجدد را بر روی ریسمان‌های مخزن ریسمان اجرا می‌کند.

موفق باشید.

اللهم عجل لولیک الفرج

مهدی محبیان

اسفند۸۹