عنوان	توسعه نرم افزار برای سیستم عامل ویندوز موبایل
عنوان اصلی	**Windows Mobile Application Development**
کلمات کلیدی	C#, Win Mobile, .NET, Dev, WinForm, GPS, Custom Control, XML, , SQL-Server, SQL-CE, AJAX, Web
مؤلف	**mstruys, dougturn**
مرجع	http://www.codeproject.com
سطح	مبتدی
مترجم	مهدی عبداللهی (http://m0911.wordpress.com)
تاریخ انتشار مجدد	۲۰ تیرماه ۱۳۸۹
تعداد صفحه	۸۸
عناوین فصل ها (فارسی)	
	۱. ایجاد نخستین برنامه ۲. شبیه ساز دستگاه و مدیریت شبیه ساز ۳. توسعه ی برنامه با WinForm ۴. افزودن کنترل سفارشی و استفاده از GPS ۵. مقدمه ای بر SQL Server CE ۶. امنیت دستگاه و نصب نرم افزار ۷. توسعه برای وب موبایل
عناوین فصل ها (انگلیسی)	
	1. Creating your first application **2. Device Emulator and Device Emulator Manager** **3. Basic WinForm App Development for Windows Mobile Devices** **4. Adding Custom Controls and Making Use of GPS Hardware** **5. Intro to using SQL Server CE** **6. Device Security and Application Deployment** **7. Mobile Web Development**

مقدمه ی مترجم

این کتاب الکترونیکی در اصل یک سری مقاله ی ۷ قسمتی برنامه نویسی ویندوز موبایل است که در اسفند ۱۳۸۸ و فروردین ۱۳۸۹ بخش به بخش ترجمه و در وبلاگم منتشر نمودم. لیکن برای راحتی کار خوانندگان ترجیح دادم که همه را در یک فایل گردآوری و باز نشر نمایم.

نویسنده گان این مجموعه مقاله از کارکنان کمپانی میکروسافت هستند. یعنی همان کمپانی که ویندوز موبایل را تولید و منتشر نموده است. از این رو مطمئنم که مطالب و محتویات این اثر برای برنامه نویسانی که می خواهند وارد دنیای ویندوز موبایل بشوند، سودمند خواهد بود.

خوشحال خواهم شد که نظر و پیشنهاد خود را از طریق آدرس وبلاگم http://m0911.wordpress.com یا به آدرس ایمیل mehdi.abd@gmail.com ارسال فرمایید.

فهرست مطالب

فصل نخست ۴

نصب ویژوال استودیو ۲۰۰۸ و ابزار های اضافی توسعه نرم افزار ۵

ایجاد نخستین برنامه ی ویندوز موبایل ۷

فصل دوم ۱۱

نگارش های مختلف شبیه ساز دستگاه ۱۲

اجرای شبیه ساز دستگاه از داخل ویژوال استودیو ۲۰۰۸ ..... ۱۳

تنظیم مشخصات شبیه ساز دستگاه ۱۴

استفاده ی هم زمان از شبیه ساز دستگاه و شبیه ساز تلفن همراه ۱۵

درست کردن اتصال داخل شبیه ساز تلفن همراه ۱۶

ارسال و دریافت پیامک ۱۷

کنترل شبیه ساز دستگاه توسط برنامه ی مدیر شبیه ساز دستگاه ۱۸

ذخیره ی تنظیمات شبیه ساز دستگاه برای استفاده های بعدی ۱۹

استفاده از شبیه ساز دستگاه برای آزمایش تنظیمات امنیتی مختلف ۲۰

خودکار سازی مدیر شبیه ساز ۲۲

فصل سوم ۲۴

دستگاه های ویندوز موبایل استاندارد ۲۵

دستگاه های ویندوز موبایل ۶ حرفه ای ۲۵

طراحی یک واسط کاربر ساده در ویندوز موبایل ۶ حرفه ای ۲۶

ایجاد نرم افزاری که بتواند با جهت های مختلف صفحه نمایش سازگار باشد ۲۷

بندکشی یا ثابت کردن کنترل ها ۲۷

طراحی برنامه ای که روی دستگاه های مختلف کار کند ۳۰

ارث بری فرم ها ۳۱

توجه به منابع سیستم ۳۴

فصل چهارم ۳۶

افزودن کنترل های سفارشی به برنامه ی خودتان ۳۷

کنترل های کاربر (سفارشی) ۳۷

کنترل های ارث بری شده ۳۹

کنترل های سفارشی ۴۱

استفاده از کنترل سفارشی داخل یک برنامه ۴۴

افزدون اطلاعات مربوط به محل جغرافیایی به برنامه ۴۵
به روز رسانی کنترل های واسط کاربر در یک برنامه ی چند پردازشی ۴۸
آزمایش برنامه های حساس به محل جغرافیایی ۴۹
فصل پنجم ۵۱
SQL Server Compact Edition ۵۲
زیر ساخت SQL Server 2005 Compact Edition ۵۲
استفاده از دیتاست های دارای نوع ۵۳
استفاده از SqlCeResultSets با نوع داده ۵۸
شکل ۳۱: SqlCeResultSet های با نوع داده ۵۹
فصل ششم ۶۵
مقدمه ۶۶
امنیت در دستگاه های ویندوز موبایل ۶۶
مجوزهای نصب نرم افزار ۶۶
تأییدیه های نرم افزار ۶۷
سطوح دسترسی نرم افزار ۶۷
تنظیمات امنیتی دستگاه ۶۷
مجوز اجرا و فایل های کتابخانه ی DLL ۶۸
آزمایش برنامه در حالت های مختلف تنظیمات امنیتی ۶۹
نصب برنامه ۷۱
به روز رسانی برنامه ها ۷۲
نصب یک برنامه ی جدید ۷۲
به روز رسانی برنامه ۷۶
شکل ۳۹: به روز رسانی خودکار در عمل ۷۶
فصل هفتم ۷۹
نرم افزار های تحت وب و پشتیبانی از ویندوز موبایل ۸۰
حالت های مختلف نمایش وب سایت ۸۱
تشخیص قابلیت های مرورگر ۸۱
تشخیص ویندوز موبایل استاندارد از پروفشنال ۸۳
فهرست تصاویر ۸۷

فصل نخست

ایجاد نخستین برنامه

مقدمه

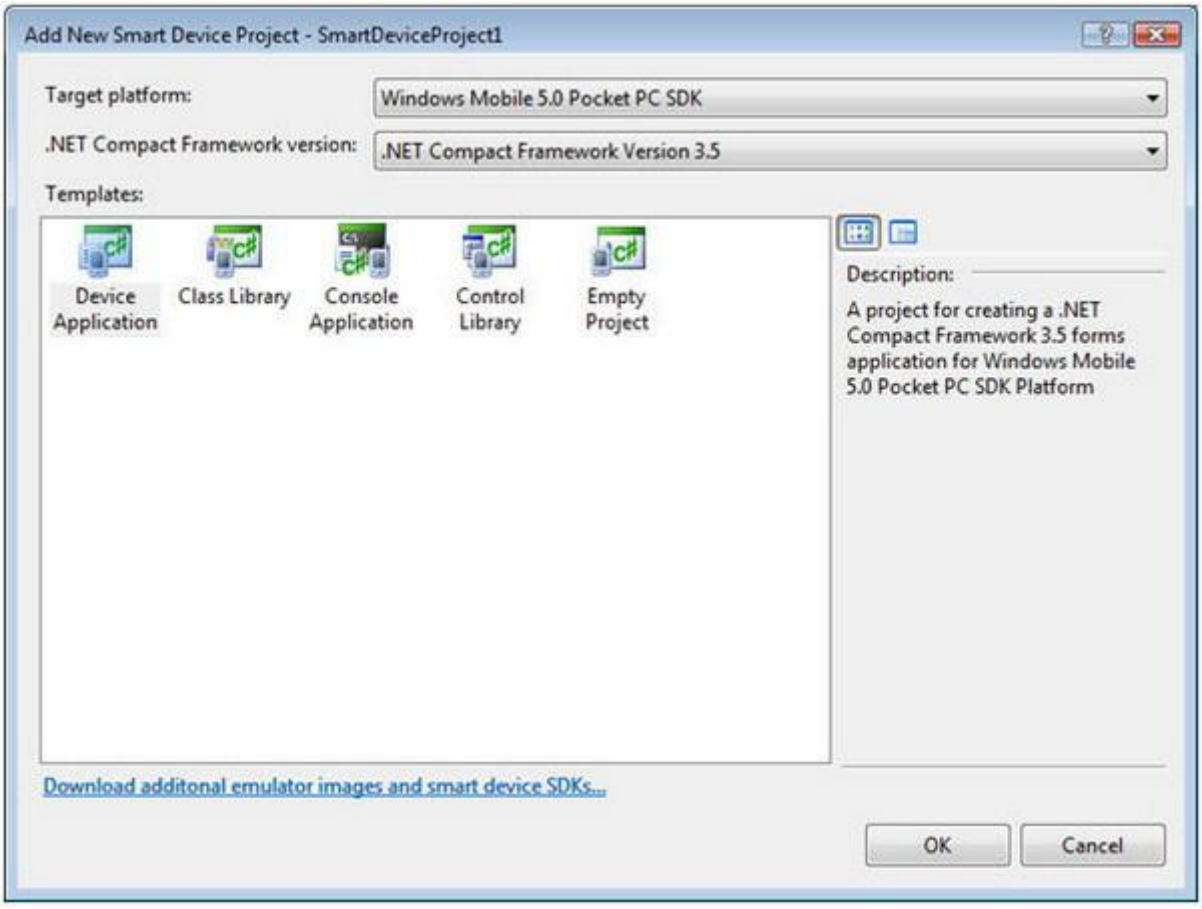
ویژوال استودیو ۲۰۰۸ نگارش حرفه ای به شما این امکان را می دهد که نرم افزارهایی به زبان های معمولی (C/C++) یا کدهای مدیریت شده (C#/Visual Basic .NET) برای دستگاه های با سیستم عامل ویندوز موبایل توسعه دهید.

در این مقاله شما یاد می گیرید که ویژوال استودیو ۲۰۰۸ را همراه با ابزارهای توسعه ی نرم افزار در ویندوز موبایل نصب کنید و نخستین برنامه ویندوز موبایل خود را برای نگارش ۶ این سیستم عامل ایجاد نمایید. به علاوه با شبیه ساز دستگاه ویندوز موبایل آشنا می شوید که امکان آزمایش برنامه های ویندوز موبایل را به شما می دهد و شما مجبور نیستید که برای این کار به طور فیزیکی، یک دستگاه واقعی ویندوز موبایل (مثلا Pocket PC یا Smart Phone) داشته باشید.

نصب ویژوال استودیو ۲۰۰۸ و ابزار های اضافی توسعه نرم افزار

اگر ویژوال استودیو ۲۰۰۸ را با تنظیمات پیش فرض نصب نمایید، گزینه ی Smart Device Development برای توسعه ی نرم افزار ویندوز موبایل فعال خواهد بود. هر دو برنامه ی ویژوال استودیو و راهنمای آن (MSDN) پس از نصب یک لینک برای به روز رسانی نمایش می دهند که توصیه می کنم با استفاده از آن آخرین به روز رسانی ها و سرویس پک ها را نصب نمایید.

پس از ورود به محیط ویژوال استودیو از منوی File – New Project گزینه ی Smart Device Project را انتخاب نمایید. این گزینه در هر دو زبان C# و Visual Basic .NET قابل دسترسی است. با فرض این که شما یک پروژه به زبان C# در ویندوز موبایل نگارش ۶ ایجاد می نمایید مطلب را ادامه می دهیم. بدین منظور در پنجره ی Project Types گزینه ی C# را باز کرده، Smart Device را انتخاب نمایید. همانند بقیه ی انواع پروژه ها نام و محل ذخیره سازی آن به طور پیش فرض تعیین می گردد و در عین حال شما خواهید توانست که نام و محل دلخواه خود را برای پروژه تان تعیین نمایید. پس از تأیید این اطلاعات پنجره ی جدید با عنوان *Add New Smart Device Project* *جهت اضافه کردن انواع مختلف پروژه های ویندوز موبایل نمایش داده می شود.* قبل از ایجاد نخستین برنامه ی ویندوز موبایل، فهرست بازشو ی Target Platform را باز کنید. اگر ویژوال استودیو ۲۰۰۸ را به صورت استاندارد نصب کرده باشید، فقط انواع محدودی از دستگاه های ویندوز موبایل را خواهید دید که Windows Mobile 6 بین آنها نیست. برای این که بتوانید در سیستم مذکور پروژه ایجاد نمایید باید ابزار توسعه ی آن یعنی Windows Mobile 6 SDK را نصب کرده باشید. این ابزار توسعه شامل مستندات، برنامه های مثال، فایل های کتابخانه ای، تصاویر محیط شبیه ساز و ابزار های لازم را به ویژول استودیو اضافه می نماید تا به کمک آنها بتوانید برنامه های مورد نظرتان را تولید کنید. این ابزار توسعه را به صورت مجزا باید نصب کرده باشید چون بعد از ارایه ویژوال استودیو به بازار، و به صورت مستقل تولید و روانه ی بازار شده اند. ابزار های توسعه (SDK) جدیدتر و متنوع تر نیز بعدها قابل دسترس خواهد بود. در پایین پنجره ی *Add New Smart Device Project* لینک ورود به صفحه ی اینترنتی Windows Mobile Developer Center قرار دارد که ابزارهای جدیدتر را می توانید از این صفحه دریافت نمایید. اگر از قبل ابزار توسعه ی Windows Mobile 6 را در محیط ویژوال استودیو ۲۰۰۵ نصب و پس از آن ۲۰۰۸ را نصب نموده اید باید ابزار توسعه را مجدد نصب کنید و گرنه در محیط ۲۰۰۸ قابل دسترس نخواهد بود.



شکل ۱: پنجره ی **New Smart Device Project**

حال اگر حداقل یکی از دو نگارش Standard و Professional ابزار توسعه ی ویندوز موبایل ۶ را نصب کرده اید می توانید نخستین برنامه ی خود را برای دستگاه با سیستم عامل ویندوز موبایل ۶ بنویسید.

ایجاد نخستین برنامه ی ویندوز موبایل

برای این برنامه باید گزینه ی Windows Mobile 6 Professional SDK را در بخش Target Platform انتخاب نمایید. البته شما هنوز می توانید برای تعداد زیادی از دستگاه های ویندوز موبایل و دستگاه های عمومی Windows CE برنامه بنویسید. با نصب هر دو نگارش ابزار توسعه ی ویندوز موبایل ۶ در ویژوال استودیو ۲۰۰۸ شما می توانید برای دستگاه های زیر برنامه بنویسید:

- Pocket PC 2003
- Windows CE (non Windows Mobile devices)
- Windows Mobile 5.0 Pocket PC
- Windows Mobile 5.0 Smartphone
- Windows Mobile 6 Professional
- Windows Mobile 6 Standard

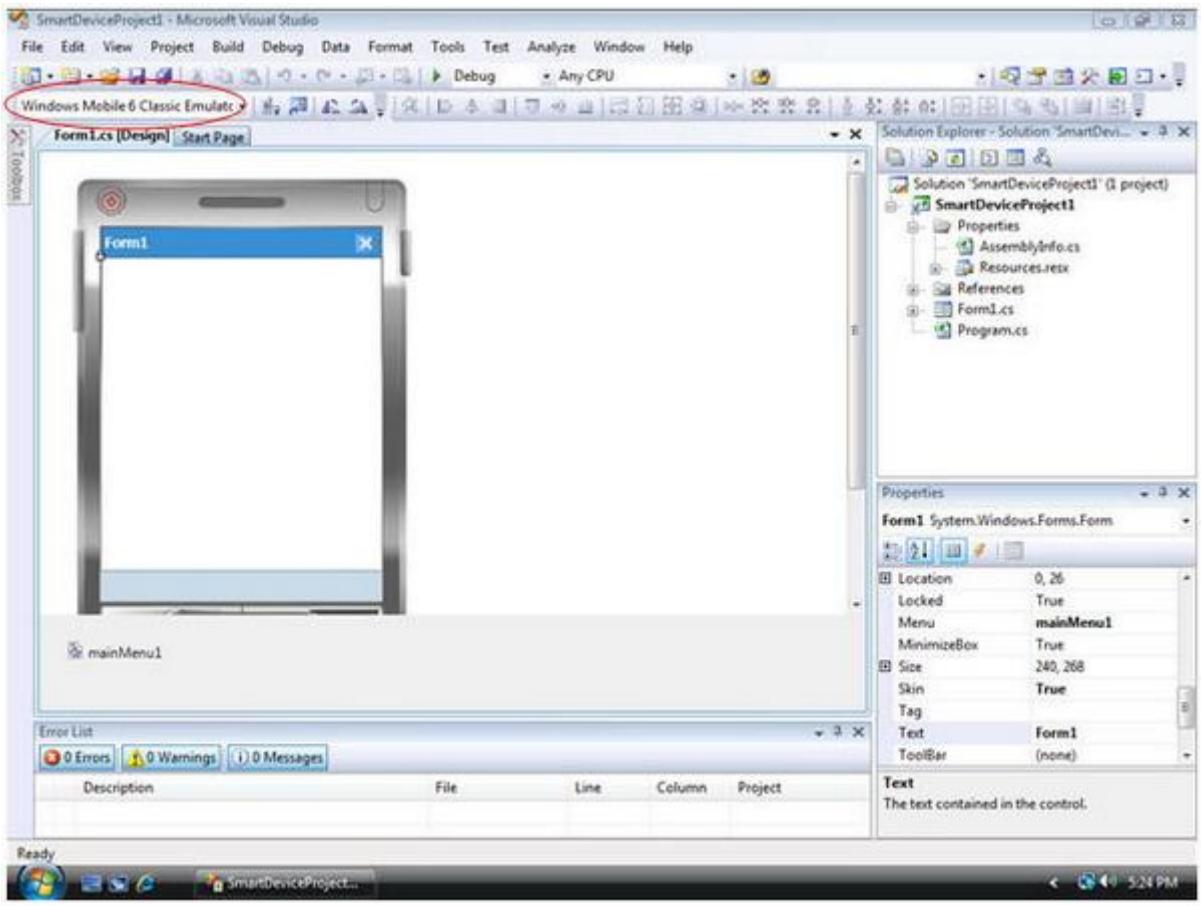
در ویژوال استودیو می توانید با دو نگارش مختلف .NET Compact Framework – که زیر مجموعه ای از فریم ورک کامل .NET. برای تولید نرم افزار دستگاه های هوشمند (Smart Device) است – برنامه بنویسید. .NET Compact Framework را در نگارش های 2.0 و 3.5 می توانید در محیط ویژوال استودیو ۲۰۰۸ استفاده نمایید.

اگر می خواهید برای دستگاه های قدیمی تر یا نگارش های قبلی .NET Compact Framework برنامه بنویسید باید ویژوال استودیو ۲۰۰۵ را نصب کنید که البته خارج از حوصله ی این مطلب می باشد. لیکن شما می توانید هم زمان نگارش های ۲۰۰۵ و ۲۰۰۸ را در یک سیستم نصب نمایید.

برای ایجاد یک پروژه ی Smart Device مراحل زیر را انجام دهید:

۱. وارد محیط ویژوال استودیو ۲۰۰۸ شوید.
۲. گزینه ی Project را از منوی File->New انتخاب نمایید.
۳. در بخش Project types ایتدا Visual C# و سپس Smart Device را انتخاب نمایید.
۴. نام پروژه و محل ذخیره سازی آن را تعیین کرده، OK را بزنید. پنجره ی دیگری مطابق شکل ۱ نمایش داده می شود که در آن، نوع پروژه (Template)، ابزار مورد استفاده (Target platform) و نگارش .NET Compact Framework. تعیین خواهید کرد.
۵. در بخش Target Platform گزینه ی Windows Mobile 6 Professional SDK انتخاب نمایید.
۶. از فهرست .NET Compact Framework version. گزینه ی .NET Compact Framework Version 3.5 را انتخاب نمایید.
۷. از فهرست Templates الگوی Device Application را انتخاب نمایید.
۸. بعد از این OK را زدید پروژه ی Smart Device solution ایجاد می شود، که شامل:
   - مرجع های مربوط به اسمبلی های مورد استفاده در برنامه.
   - پرونده ی *AssemblyInfo.cs که اطلاعات شرکت، محصول نرم افزاری و نگارش آن را در این پرونده ذخیره می نمایید.*
   - *پرونده ی Form1.cs* که حاوی کد به زبان ویژوال C# برای واسط کاربری اولیه ی برنامه است.
   - پرونده ی *Program.cs حاوی کد به زبان ویژوال C#* که در واقع نقطه ی شروع اجرای برنامه ی شما است.

اگر همه چیز خوب پیش رفته باشد، شما محیط طراحی فرم ویژوال استودیو ۲۰۰۸ را به صورت زیر خواهید دید.



شکل ۲: محیط طراحی فرم ویژوال استودیو ۲۰۰۸

در شکل ۲ چند بخش مهم از محیط طراحی ویژوال استودیو ۲۰۰۸ را می بینید. بخش بزرگ سمت چپ صفحه که شکل یک دستگاه موبایل با سیستم عامل ویندوز را نشان می دهد و در واقع به جای طراح فرم در برنامه های معمولی است. شما می توانید کنترل های مورد نیاز واسط کاربری را از جعبه ابزار آن انتخاب کرده، داخل Form1 در محل مورد نظر قرار دهید.

پنجره ی سمت راست بالا به نام Solution Explorer است که تمام پروژه ها –که بخضی از برنامه ی شما هستند - و فایل های داخل هر کدام از آنها را به صورت نمای درختی (tree view) نشان می دهد. در بخش پایین پنجره ی Solution Explorer شما پنجره ی با عنوان Properties را می بینید که در آن می توانید مشخصات هر کدام از کنترل های انتخاب شده ی واسط کاربری را ببینید و تغییر دهید.

نکته: اگر پنجره ی Properties را در صفحه ویژوال استودیو ندارید می توانید گزینه ی Properties Window را در منوی View فعال نمایید.

بخش مهم دیگر در محیط ویژوال استودیو ۲۰۰۸ فهرست باز شوی Target Device است که در شکل ۲ با خط قرمز بیضی شکل مشخص شده است. در اینجا شما می توانید تعیین کنید که برنامه تان را برای اجرا در کدام دستگاه یا کدام شبیه ساز توسعه می دهید.

برای این که امکانات عملیاتی به برنامه تان اضافه کنید مراحل زیر را انجام دهید.

۱. یک کنترل دکمه از جعبه ابزار بکشید و روی Form1 قرار دهید.
۲. در پنجره ی Properties مشخصه ی Name را از مقدار پیش فرض button1 به btnClickMe تغییر دهید.
۳. در همان پنجره مشخصه ی Text را از button1 به Click ME! تغییر دهید.
۴. یک کنترل برچسب از جعبه ابزار بکشید و روی Form1 قرار دهید و طوری تغییر اندازه بدهید که تمام عرض Form1 را بپوشاند.
۵. در پنجره ی Properties مشخصه ی Name را از مقدار پیش فرض label1 به lblInfo تغییر دهید.
۶. در همان پنجره مقدار مشخصه ی Text را خالی کنید.
۷. برای این که مشخصات Form1 در پنجره ی Properties ظاهر شود روی محل دلخواهی از Form1 کلیک کنید.
۸. مقدار مشخصه ی MinimizeBox را False قرار دهید تا با کلیک روی دکمه ی OK – که در گوشه ی سمت راست بالای Form1 هست – اجرای برنامه تمام شود.
۹. روی دکمه ی Click ME! دابل کلیک کنید تا مدیر رویداد Click را به دکمه اضافه نمایید.
۱۰. در محلی از ویرایشگر کد که مکان نما قرار دارد، کد زیر را اضافه نمایید.

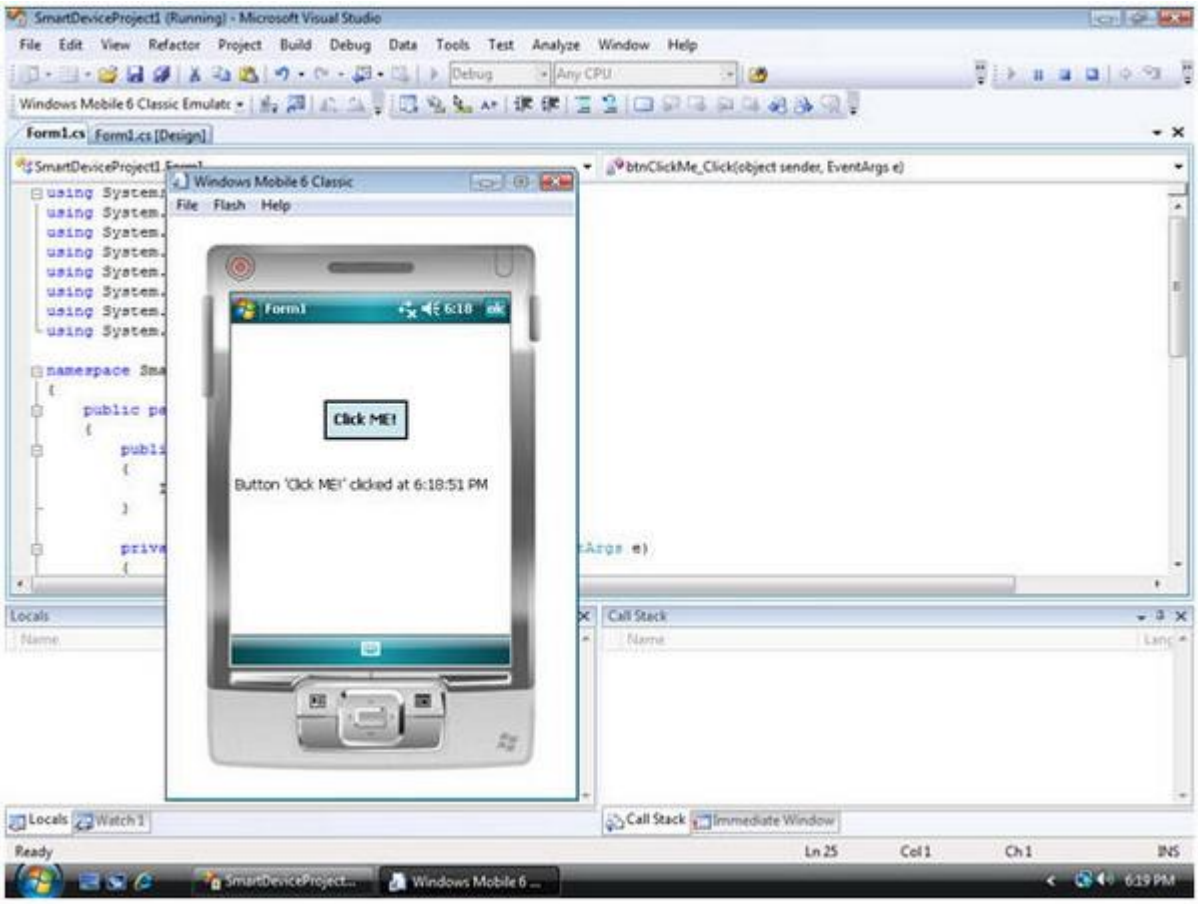
```
lblInfo.Text = "Button '" +
               btnClickMe.Text +
               "' clicked at " +
               DateTime.Now.ToLongTimeString();
```

نکته: شما حین نوشتن کد، قابلیت نمایش رنگی کلمات کلیدی و ... محیط ویرایشگر کد را – که هر کدام از دستورها و عبارات تایپ شده را برای راحتی تشخیص درست بودن آن، به رنگ خاصی نمایش می دهد – مشاهده خواهید کرد. همچنین Intellisense نیز شما را در وارد کردن دستور العمل ها کمک خواهد کرد.

برای آماده سازی و اجرای برنامه تان مراحل زیر را انجام دهید.

۱. از منوی Build گزینه ی Build Solution را انتخاب کنید. با فرض این که هیچ خطایی در تایپ دستورهای برنامه نداشته باشید پیغام Build succeeded در نوار وضعیت ویژوال استودیو ۲۰۰۸ نمایش داده خواهد شد.
۲. بررسی کنید که گزینه ی Windows Mobile 6 Classic Emulator در فهرست بازشو ی Target Device انتخاب شده باشد.
۳. از منوی Debug گزینه ی Start Debugging را انتخاب کنید.

بعد از چند لحظه شبیه ساز کلاسیک ویندوز موبایل نگارش ۶ اجرا می شود. در صورت لزوم .NET Compact Framework 3.5 در شبیه ساز کپی و نصب خواهد شد. در مرحله ی پایانی، برنامه ی شما نصب و اجرا خواهد شد.



شکل ۳: نخستین برنامه شما در حال اجرا در شبیه ساز دستگاه ویندوز موبایل

برای آزمایش اجرای برنامه تان مراحل زیر را انجام دهید.

۱. وارد برنامه –که در شبیه ساز در حال اجرا است – بشوید و دکمه ی Click ME! را کلیک نمایید. متنی را که در برچسب روی فرم ظاهر می شود، خواهید دید.
۲. با هر بار کلیک روی دکمه می بینید که ساعت نمایش داده شده روی برچسب به روز می شود.
۳. برای خروج از برنامه دکمه ی OK را در گوشه ی سمت راست بالای صفحه ی برنامه کلیک کنید.
۴. با انتخاب گزینه ی Save State and Exit از منوی File برنامه ی شبیه ساز، آن را ببندید.
۵. از ویژوال استودیو ۲۰۰۸ خارج شوید.

فصل دوم

شبیه ساز دستگاه و مدیریت شبیه ساز

مقدمه

اگر شما توسعه دهنده ی برنامه های ویندوز موبایل باشید حتما با شبیه ساز دستگاه آشنا هستید. این یک ابزار رایگان برای آزمایش اجرای برنامه های تان بر روی انواع مختلف دستگاه های با سیستم عامل ویندوز موبایل با مدل های مختلف است؛ بدون این که نیاز به تهیه ی دستگاه های واقعی باشد.

در این مقاله شما با نحوه ی کار شبیه ساز دستگاه برای آزمایش برنامه های خود آشنا می شوید و این که چگونه شبیه ساز را تنظیم نمایید تا دقیقا بتوانید عملکرد ابزار را کنترل کنید. به علاوه چگونگی استفاده از شبیه ساز را به صورت ترکیبی با تلفن همراه نیز خواهید آموخت که بتوانید تماس های تلفنی و ارسال و دریافت پیامک را آزمایش نمایید. در نهایت خواهید دید که چگونه می توانید به کمک برنامه نویسی، از مدیر شبیه ساز برای کنترل شبیه ساز های مختلف از داخل یک برنامه ی دسکتاپ استفاده کنید.

نگارش های مختلف شبیه ساز دستگاه

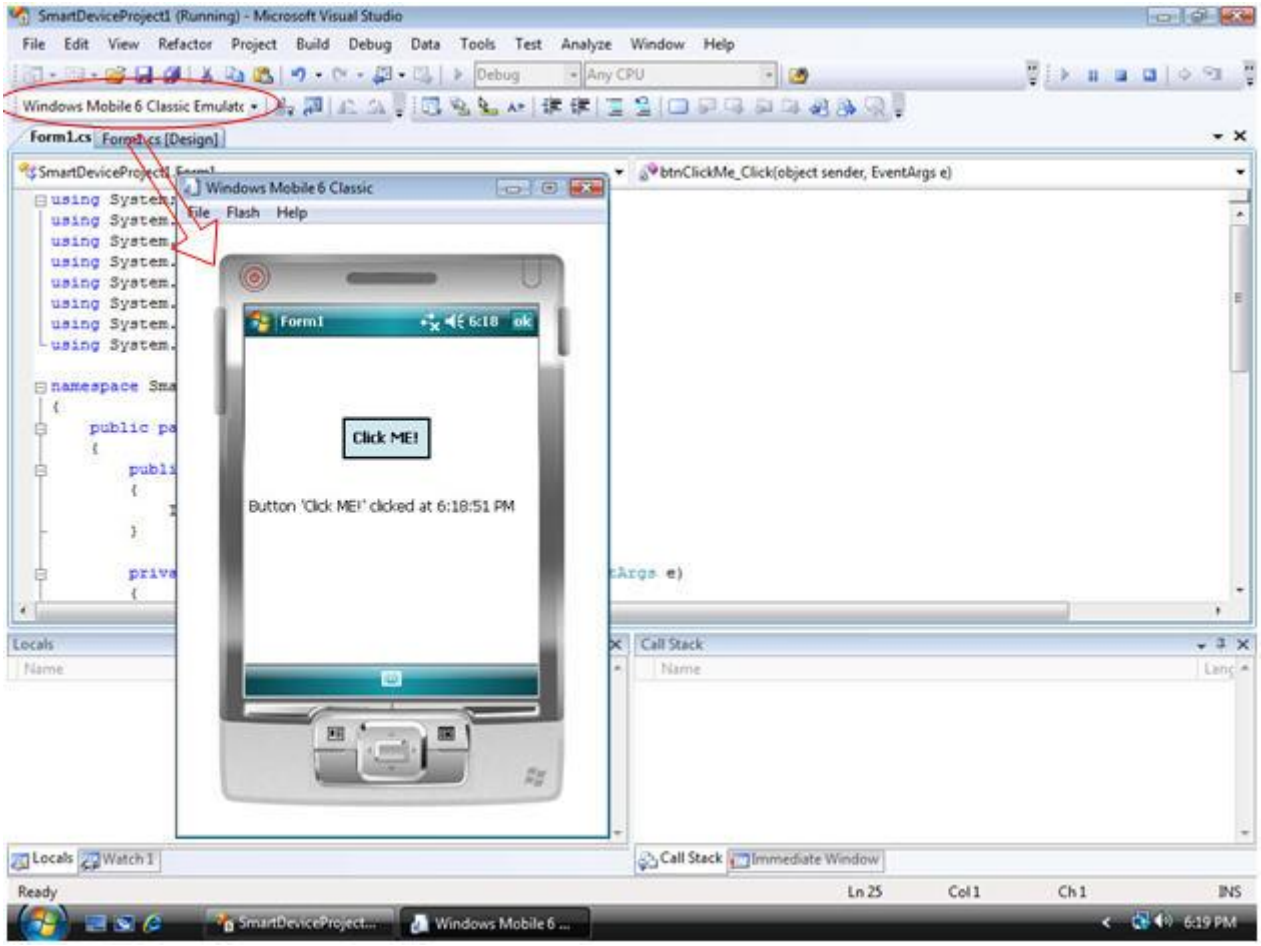
نگارش ۱.۰ شبیه ساز دستگاه همراه با ویژوال استودیو ۲۰۰۵ عرضه شد. این ابزار به شما امکان می داد که بدون نیاز به دستگاه واقعی ویندوز موبایل برنامه های خود را آزمایش کنید. به محض ارایه ی نگارش ۱.۰ نگارش ۲.۰ شبیه ساز نیز همراه با ابزار توسعه ی ویندوز موبایل ۶ عرضه شد که عملکرد و کارایی آن نسبت به نگارش ۱.۰ گسترش یافت. در بخشی از مراحل نصب ابزار توسعه ی ویندوز موبایل ۶، نگارش ۲.۰ شبیه ساز دستگاه نیز جایگزین نگارش ۱.۰ می گردد. در واقع تمامی پروژه های ویندوز موبایل شما که با ویژوال استودیو ۲۰۰۵ توسعه داده اید از نگارش جدید استفاده خواهند کرد. حتی اگر شما پلاتفرم مقصد را ویندوز موبایل ۶ تعیین نکنید و بخواهید همچنان از نگارش ۲۰۰۵ استفاده نمایید، باز هم بهتر است که حداقل یک ابزار توسعه ویندوز موبایل ۶ را نصب کنید تا نه فقط از سرعت بالاتر بلکه از پیشرفت های آن مانند امکان تنظیمات جزء به جزء برای بخش های خاص دستگاه، یا امکان شبیه سازی اتصال بی سیم استفاده نمایید.

نگارش ۰.۳ آخرین نگارش شبیه ساز دستگاه (در زمان نگارش این مقاله) است که همراه با ویژوال استودیو ۲۰۰۸ عرضه شده است و به طور خودکار نگارش های قبلی نصب شده روی سیستم شما را با نگارش جدید جایگزین می نماید. نگارش ۰.۳، علاوه بر بهبود کارآیی نسبت به نگارش ۰.۲، از واسط های COM مدیر شبیه ساز دستگاه استفاده می کند. به علاوه امکان کنترل شبیه ساز ها را از طریق برنامه نویسی برای موارد زیر فراهم می سازد:

- اجرای نمونه (instance)
- خودکارسازی اجرای شبیه سازها
- تغییر تنظیمات شبیه ساز

اجرای شبیه ساز دستگاه از داخل ویژوال استودیو ۲۰۰۸

چند روش برای اجرای شبیه ساز از داخل ویژوال استودیو ۲۰۰۸ وجود دارد. کافی است که یکی از انواع شبیه ساز ها را که منطبق بر SDK ویژه ی پلاتفرم مقصد برنامه ی شما است، از داخل ویژوال استودیو ۲۰۰۸ انتخاب کنید و برنامه تان را اجرا کنید. قبل از اجرای برنامه تان روی شبیه ساز انتخاب شده، برنامه تان کامپایل می شود و شبیه ساز نیز اگر در حال اجرا نباشد، بالا می آید و به ویژوال استودیو ۲۰۰۸ متصل می شود. بعد از ساخته شدن برنامه ی اجرایی در صورت لزوم SQL Server CE همراه با نگارش درست .NET Compact Framework به طور خودکار در پیبه ساز دستگاه اجرا خواهد شد.



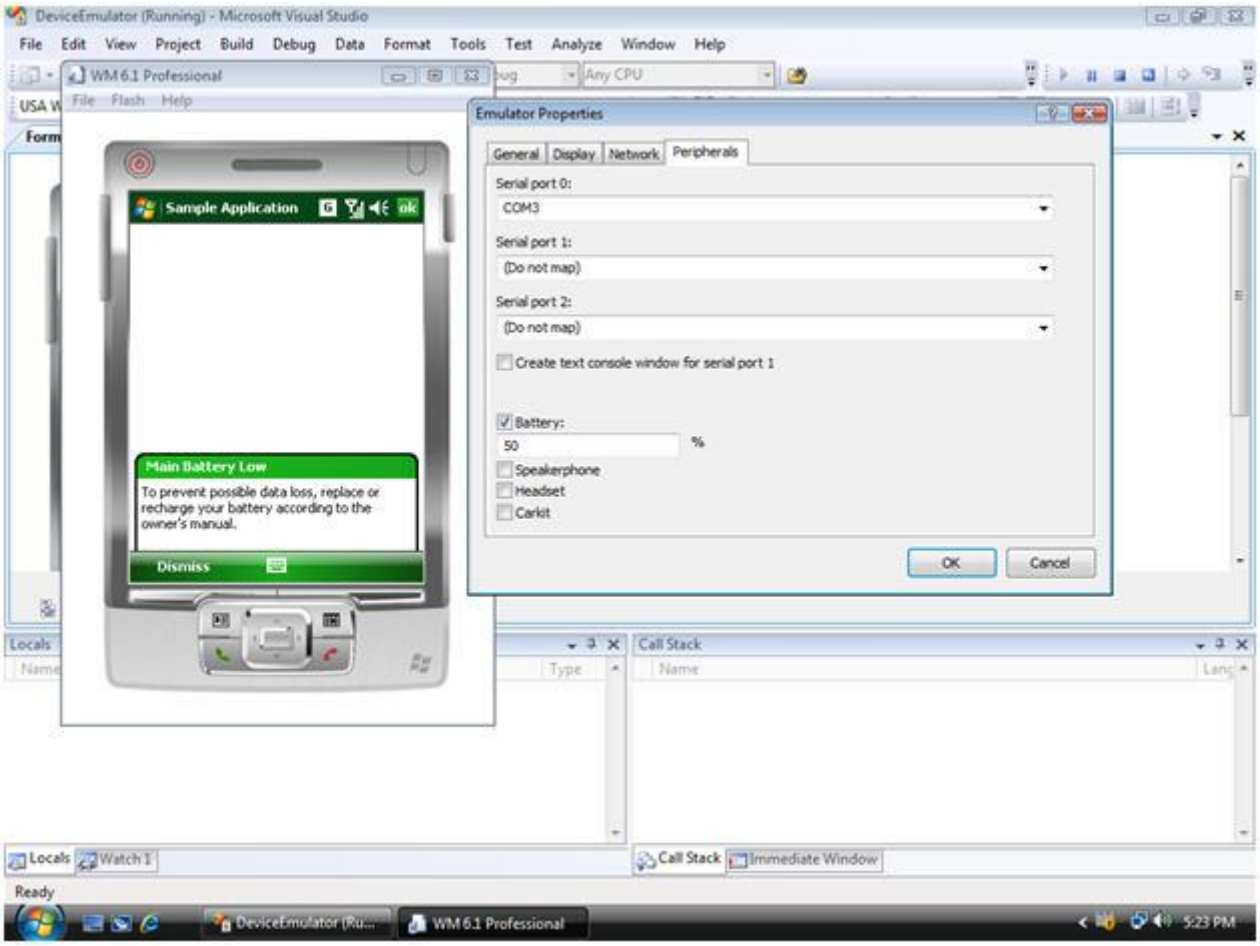
شکل ۴: برنامه ی ویندوز موبایل در حال اجرا روی شبیه ساز دستگاه

تنظیم مشخصات شبیه ساز دستگاه

اغلب اوقات ممکن است بخواهید برنامه ویندوز موبایل تان را با تنظیمات مختلفی از دستگاه آزمایش کنید. برای این حالت ها شبیه ساز دستگاه این امکان را به شما می دهد که تجهیزات مختلفی را شبیه سازی نمایید. مثلا برای میزان شارژ مشخصی از باتری که بخواهید برنامه تان را در حالت باتری ضعیف امتحان کنید. یا این که یک کارت شبکه ی مجازی را به کارت شبکه ی فیزیکی خود که روی کامپیوتر تان هست متصل نمایید و همچنین پورت ارتباطی (COM) مجازی شبیه ساز را به پورت ارتباطی فیزیکی روی کامپیوترتان متصل نمایید. یا این که پوشه ی خاصی را بین کامپیوتر خود (که روی آن برنامه را توسعه می دهید) و شبیه ساز (از طریق پنجره ی مشخصات) به اشتراک بگذارید. پوشه ی اشتراکی به صورت کارت حافظه (Storage Card) داخل شبیه ساز دستگاه مشاهده خواهد شد. با استفاده از پوشه اشتراکی می توانید فایل ها را بین شبیه ساز و کامپیوتری که دارید برنامه تان را روی آن توسعه می دهید مبادله کنید.

برای نمایش پنجره ی مشخصات شبیه ساز از منوی File شبیه ساز گزینه ی Configure را انتخاب نمایید.

با کلیک روی گزینه ی Options در منوی Tools ویژوال استودیو ۲۰۰۸ نیز می توانید همین پنجره ی مشخصات را مشاهده نمایید. ابتدا در پنجره ی Options گزینه ی Device Tools را باز کرده، Devices را کلیک کنید. در پنجره ای که باز می شود شبیه ساز مورد نظر تان را انتخاب و روی Properties کلیک نمایید. با کلیک روی Emulator Options پنجره ی Emulator Properties باز خواهد شد.



شکل ۵: تنظیم مشخصات شبیه ساز

استفاده ی هم زمان از شبیه ساز دستگاه و شبیه ساز تلفن همراه

موارد بسیاری پیش می آید که بخواهید اجرای نرم افزار تان را در تلفن همراه آزمایش کنید. مثلا نرم افزار شما ممکن است شماره گیری تلفن یا ارسال و دریافت پیامک را انجام دهد. برای این موراد شبیه ساز دستگاه بسیار مفید خواهد بود چرا که کمک می کند تا بدون نیاز به یک تلفن همراه واقعی تمامی حالت های آن را آزمایش نمایید. شبیه ساز تلفن در نگارش ۰.۲ و بالاتر قابل دسترس است. شبیه ساز تلفن همراه را از منوی **Start/All Programs / Windows Mobile 6 SDK / Tools / Cellular Emulator** اجرا نمایید. پیش از آن که شبیه ساز دستگاه را همراه با شبیه ساز تلفن همراه استفاده نمایید، باید پورت سریال شماره ۰ شبیه ساز دستگاه را به پورت سریالی که در نوار وضعیت برنامه ی شبیه ساز تلفن همراه نمایش داده می شود متصل نمایید. در شکل ۳ شما پورت سریال شماره ۰ را در صفحه ی مشخصات برنامه ی شبیه ساز می بینید که برای ارتباط با برنامه ی شبیه ساز تلفن همراه به پورت COM3 متصل شده است. پس از اتصال پورت سریال باید شبیه ساز دستگاه را راه اندازی مجدد (reset) کنید تا شبیه ساز تلفن همراه کار کند.

شکل ۶: اتصال شبیه ساز تلفن همراه به شبیه ساز دستگاه

حال می توانید:

- گرفتن شماره تلفن
- دریافت تماس تلفنی
- استفاده از اتصال GPRS داخلی شبیه ساز دستگاه برای اتصال به شبکه
- ارسال و دریافت پیامک
- انتخاب یکی از دو حالت شبکه 2G و 3G
- پردازش فرمان های سطح پایین مودم

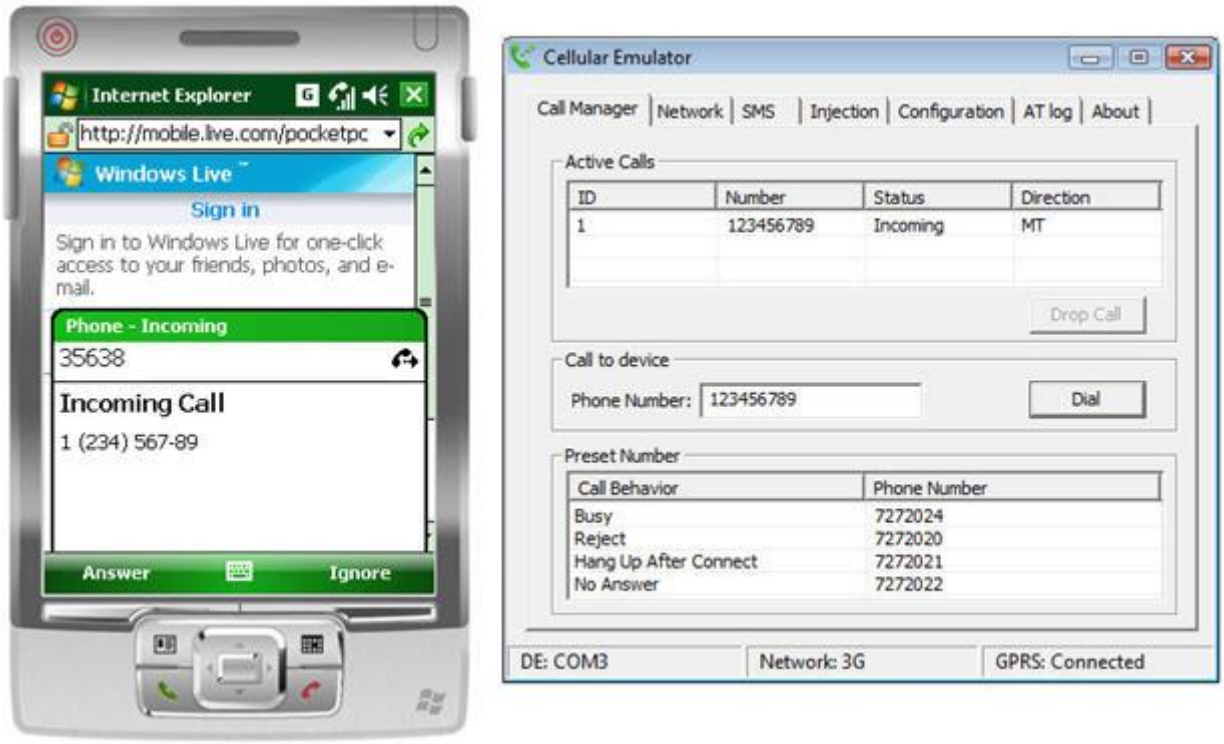
درست کردن اتصال داخل شبیه ساز تلفن همراه

پس از اتصال شبیه ساز دستگاه به شبیه ساز تلفن همراه، شما آزمایش های متنوعی را با آن می توانید انجام دهید. شما بلافاصله می توانید تماس تلفنی و همچنین پیامک از / به شبیه ساز دستگاه داشته باشید. به علاوه می توانید اتصال شبکه ی GPRS از طریق شبیه ساز دستگاه درست کنید. برای یادگیری درست کردن اتصال شبکه در شبیه ساز دستگاه به آدرس زیر مراجعه کنید:

http://msdn.microsoft.com/en-us/library/bb158505.aspx

در شکل ۴ چند اتصال دیتا که بین شبیه ساز دستگاه و شبیه ساز تلفن همراه فعال هستند، می بینید. در پس زمینه ی شبیه ساز دستگاه، برنامه ی اینترنت اکسپلورر را می بینید که یک وب سایت را باز کرده است. اطلاعاتی که از طریق اتصال GPRS رسیده است، طبق آن چه در نوار وضعیت شبیه ساز دستگاه می بینید، به شما نشان می دهد که GPRS متصل شده است. شبیه ساز تلفن همراه، اتصال GPRS را با استفاده از اتصال شبکه ای کامپیوتری که روی آن اجرا شده است، فراهم می کند.

در شکل ۴ شما برنامه ی مدیر تماس شبیه ساز تلفن همراه را می بینید که با شبیه ساز دستگاه، تماس گرفته است. در جعبه متن Phone Number می توانید هر شماره تلفنی را وارد نمایید و دکمه ی Dial را بزنید. بلافاصله بعد از این کار، داخل شبیه ساز دستگاه پیغام تماس ورودی (incoming call) را خواهید دید و همان شماره ای که شما از داخل شبیه ساز تلفن همراه گرفته اید نمایش داده خواهد شد.



شکل ۷: شبیه ساز دستگاه و شبیه ساز تلفن همراه در حال اجرا

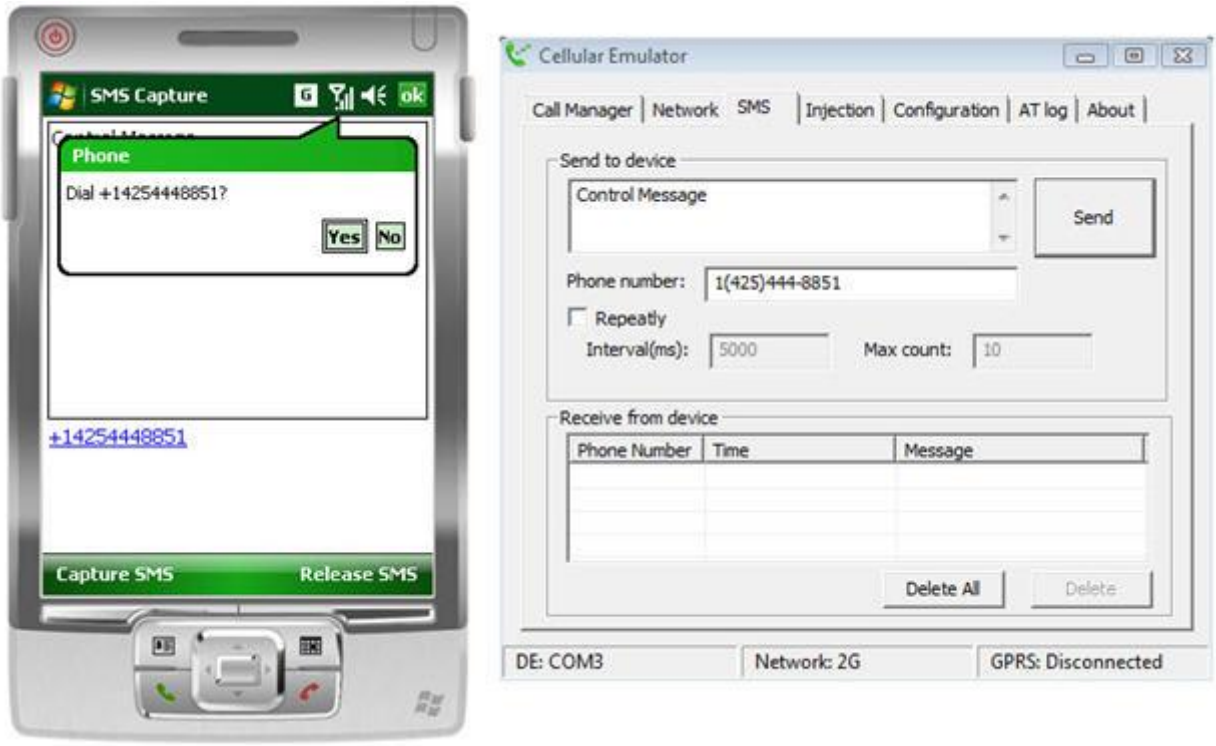
ارسال و دریافت پیامک

یک قابلیت مهم شبیه ساز تلفن همراه ارسال / دریافت پیامک به / از شبیه ساز دستگاه است. توجه کندی که صرف دریافت پیامک از طریق شبیه ساز دستگاه مهم نیست. به محض دریافت پیامک، پس از اطلاع دادن به کاربر پیامک دریافتی در پوشه ی inbox شبیه ساز دستگاه ذخیره می گردد. در عین حال شما نیز می توانید پیامک را از طریق برنامه ی خودتان دریافت کنید. همین قابلیت به شما امکانات متنوعی می دهد به خصوص زمانی که شما بتوانید یک برنامه را که روی یک دستگاه ویندوز موبایل اجرا می شود، از راه دور کنترل کنید. یا برنامه ی شما پیامک را به طور انحصاری دریافت کرده (به دام انداخته)، راه ورود آن به سیستم را مسدود نماید یا ابتدا آن را مسدود و سپس به پوشه ی inbox داخل دستگاه ارسال نماید.

به ویژه در حالتی که شما از پیامک برای کنترل برنامه تان استفاده می کنید، ممکن اس پیامک های بسیاری را به دستگاه ارسال نمایید. ترکیب شبیه ساز دستگاه و شبیه ساز تلفن همراه برای چنین حالت های تست برنامه هیچ هزینه ای نخواهد داشت و شما به شبکه ی فیزیکی تلفن همراه یا مثلا دو سیم کارت و دو عدد گوشی و در نهایت هزینه ی پیامک به اپراتور تلفن همراه، نیاز نخواهید داشت. به دلیل محدودیت این مقاله برای توضیح شبیه ساز دستگاه، ما به جزئیات نحوه ی دریافت پیامک از طریق برنامه نویسی اشاره نمی کنیم. برای اطلاعات بیشتر به مرجع MSDN آدرس زیر مراجعه کنید:

http://msdn.microsoft.com/en-us/library/microsoft.windowsmobile.pocketoutlook.messageinterception.messageinterceptor.aspx

در شکل ۵ شما یک نمونه برنامه می بینید که پیامک هایی را که با عبارت Control Message آغاز شوند، جدا و دریافت می نماید. هر بار که پیامک توسط دستگاه – که در حال اجرای برنامه ی SMS Capture است – دریافت می شود، ابتدا توسط برنامه بررسی می شود. برنامه متن پیامک را در یک جعبه متن نمایش می دهد و مشخصه ی Text یک کنترل LinkLabel را برابر با شماره تلفن ارسال کننده ی پیامک قرار می دهد. کاربر خواهد توانست به سادگی با کلیک روی شماره تلفن نمایش داده شده، با فرستنده ی پیامک تماس تلفنی بگیرد.



شکل ۸: دریافت پیامک داخل یک برنامه

کنترل شبیه ساز دستگاه توسط برنامه ی مدیر شبیه ساز دستگاه

ویژوال استودیو ۲۰۰۸ ابزاری به نام «Device Emulator Manager» دارد که برای کنترل شبیه ساز های دستگاه - که روی کامپیوتر میزبان نصب شده است- برای اجرای این برنامه، از منوی Tools گزینه ی Device Emulator Manager را انتخاب کنید. برنامه ی مدیر شبیه ساز فهرستی از شبیه ساز های دستگاه ها را به صورت نمودار درختی به شما نشان می دهد و شما می توانید به شبیه ساز دستگاه مورد نظر خود متصل شده، دستگاهی را حذف یا اضافه نموده، در نهایت دستگاه را خاموش کنید. اضافه کردن شبیه ساز دستگاه امکان بسیار مفیدی است، چون این قابلیت را فراهم می کند شبیه ساز دستگاه توسط برنامه ی (ActiveSync) در ویندوز ایکس پی و Windows Mobile Device Center در ویندوز ویستا و ۷ به کامپیوتر میزبان متصل گردد. این برنامه به محض اجرا امکان می دهد که شبیه ساز دستگاه محتویات خود را همسان سازی (Syncronize) کند، به اینترنت متصل گردد یا به وسیله ی فایل CAB. برنامه ای را از کامپیوتر میزبان روی شبیه ساز دستگاه نصب کند.

نکته: شما باید برنامه ی ActiveSync یا Windows Mobile Device Center را برای اتصال شبیه ساز دستگاه به کامپیوتر، در سیستم خود نصب کرده باشید. در ضمن اتصال از طریق DMA (دسترسی مستقیم به حافظه = Direct Memory Access) را نیز باید فعال نمایید.

(ر.ک http://msdn.microsoft.com/en-us/library/aa188173.aspx)

شکل ۹: مدیر شبیه ساز در حال اجرای یک شبیه ساز دستگاه

در شکل ۶ ویندوز موبایل ۱.۶ را می بینید که در یک صفحه ی مربعی داخل مدیر شبیه ساز اجرا شده است.پس از اجرای شبیه ساز، برنامه ی مدیر شبیه ساز برای اتصال به آن به کار می رود. برای کنترل شبیه ساز ها از داخل مدیر شبیه ساز باید روی شبیه ساز مورد نظر تان از لیست نمایش درختی – که همه ی شبیه ساز های نصب شده را نشان می دهد – راست کلیک کنید و گزینه ی مورد نظر تان را از منوی Actions روی شبیه ساز در حال اجرا انتخاب نمایید. به محض اتصال به شبیه ساز دستگاه خواهید دید که برنامه ی ActiveSync یا Windows Mobile Device Center بالا می آیند.

ذخیره ی تنظیمات شبیه ساز دستگاه برای استفاده های بعدی

با برنامه ی مدیر شبیه ساز این امکان هست که شما تنظیمات خاص مورد نظرتان را برای هر نوع شبیه ساز دستگاه ذخیره نمایید. مثلا یک شبیه ساز ایجاد می کنید که همیشه با باتری ۵۰٪ و کارت حافظه ای که روی یک پوشه ی خاص روی کامپیوتر میزبان تنظیم شده است، بالا بیاید. بدین منظور در برنامه ی مدیر شبیه ساز یکی از شبیه ساز های نصب شده را انتخاب و پس از راست کلیک روی آن گزینه ی Connect را بزنید. به هنگام اجرای شبیه ساز می توانید تنظیمات آن را تغییر دهید و توسط گزینه ی Save As از منوی راست کلیک داخل مدیر شبیه ساز تنظیمات جدید را ذخیره نمایید.این تنظیمات ذخیره شده به صورت یک نام شبیه ساز جدید داخل مدیر شبیه ساز دیده خواهند شد. در نمودار درختی **My Device Emulators** داخل مدیر شبیه ساز می توانید این شبیه ساز دستگاه ها را ببینید. با این کار صرفا یک فایل حاوی تنظیمات بر مبنای نوع خاص شبیه ساز مورد نظر شما ایجاد می شود و با هر بار ایجاد یک شبیه ساز با تنظیمات جدید حافظه ی چندانی برای آن روی هارد دیسک شما اختصاص نمی یابد.

شکل ۱۰: تنظیمات شبیه ساز دستگاه

در شکل ۷ نموه ای از تنظیمات شبیه ساز دستگاه را می بینید که یک کارت حافظه در مسیر *C:\Users\Public\Documents* روی کامپیوتر میزبان برای آن تعریف شده است و حالت باتری ۵۰٪ را نیز به هنگام اجرا شبیه سازی کرده است. هر بار که این شبیه ساز خاص اجرا شود همین تنظیمات از ابتدا برقرار خواهند بود.

استفاده از شبیه ساز دستگاه برای آزمایش تنظیمات امنیتی مختلف

هنگامی که از شبیه ساز دستگاه برای آزمایش برنامه های ویندوز موبایل تان استفاده می کنید با فرض اتصال ویژوال استودیو به شبیه ساز دستگاه هیچ مانعی برای نصب و اجرای آنها نخواهید داشت. شبیه ساز مانند یک ویندوز موبایل کامل و بدون محدودیت امنیتی برای شما کار خواهد کرد. بسته به تنظیمات امنیتی روی دستگاه واقعی ویندوز موبایل ممکن است شما ممکن است محدودیت های امنیتی برای دسترسی به آن داشته باشید. با استفاده از ویژوال استودیو ۲۰۰۸ خواهید توانست تنظیمات امنیتی مورد نظرتان را روی شبیه ساز اعمال نمایید. لیکن توضیح این که چه تنظیمات امنیتی را می توانید در ویندوز موبایل اعمال کنید خارج از حوصله ی این مقاله است. از طریق ادرس زیر در سایت میکروسافت می توانید به تفصیل تنظیمات امنیتی ویندوز موبایل را ببینید:

http://msdn.microsoft.com/en-us/library/bb416353.aspx

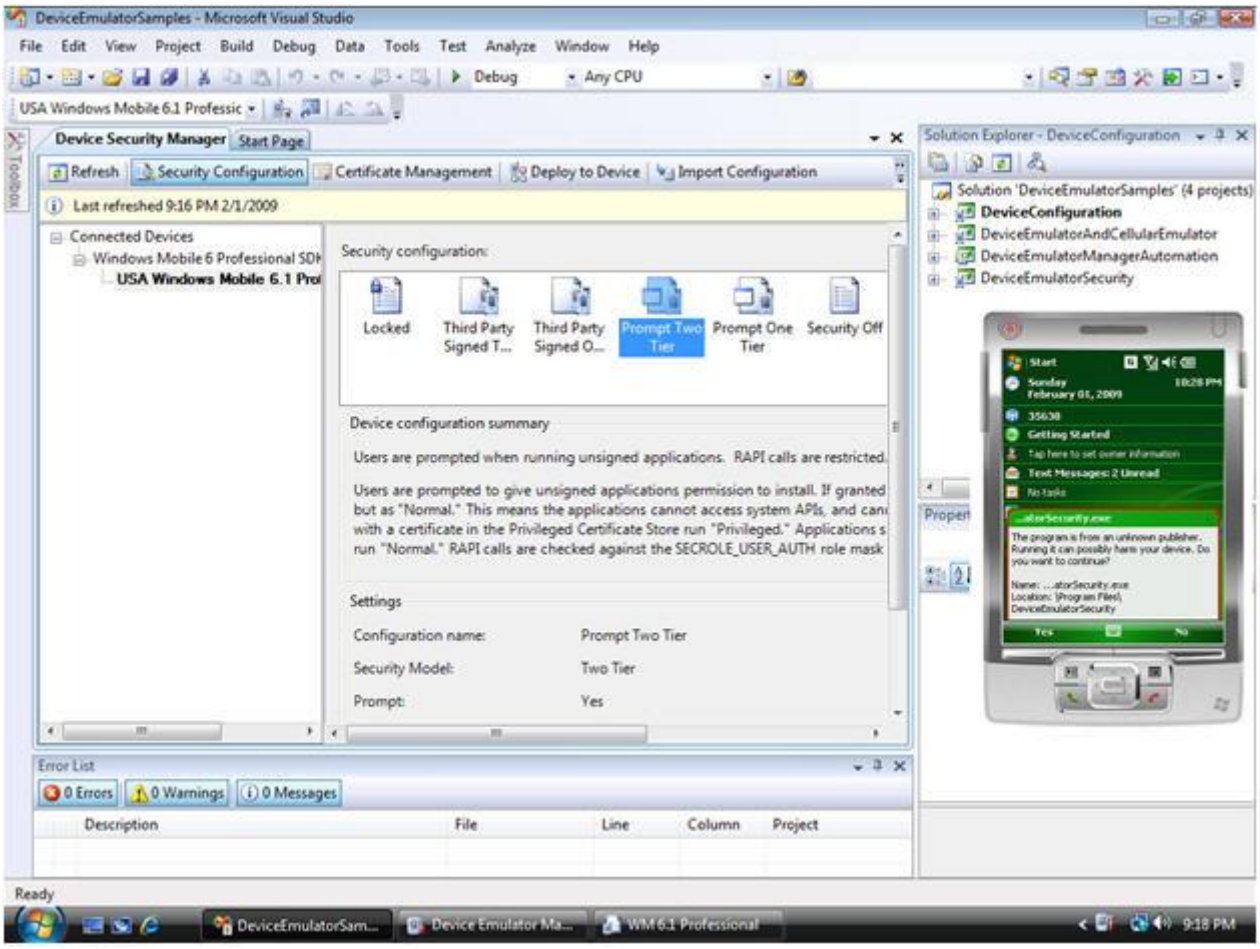
(به زودی محتویات این صفحه ترجمه خواهد شد. مترجم)

اعمال تنظیمات امنیتی خاص در شبیه ساز دستگاه سرراست و آسان است. بعد از اجرای یک شبیه ساز دستگاه می توانید تنظیمات امنیتی فعلی آن را استخراج نمایید. بدین منظور از منوی Tools ویژوال استودیو ۲۰۰۸ گزینه ی Device Security Manager را انتخاب نمایید. پس از این، تنظیمات امنیتی پیش فرض محیط ویژوال استودیو ۲۰۰۸ را خواهید دید. توسط Device Security Manager می توانید تنظیمات امنیتی شبیه ساز دستگاه در حال اجرا و همچنین تأییدیه های نصب شده را ببینید.

نکته: از برنامه ی Device Security Manager جهت استخراج و بررسی تنظیمات امنیتی دستگاه واقعی (فیزیکی) ویندوز موبایل هم می توانید استفاده کنید.

برای اعمال تنظیمات امنیتی جدید در یک شبیه ساز دستگاه مراحل زیر را انجام دهید:

۱. از برگه ی Device Security Manager گزینه ی Security Configuration را بزنید
۲. از فهرست Security configuration تنظیمات مورد نظرتان را انتخاب کنید.
۳. روی برگه ی Deploy to Device داخل Device Security Manager کلیک کنید تا تنظیمات امنیتی مورد نظرتان روی شبیه ساز اعمال شود.



شکل ۱۱: مدیر امنیت دستگاه در حال اجرا از داخل شبیه ساز دستگاه

خودکار سازی مدیر شبیه ساز

کارآیی مدیر شبیه ساز دستگاه به صورت یک سری اشیای COM (Component Object Model) است که می توانیم با آن ها برنامه های دسکتاپ ویندوز برای کنترل مدیر شبیه ساز بنویسیم. به عنوان مثل شما می توانید به زبان C# برنامه ای بنویسید که به یک شبیه ساز خاص متصل شده آن را اجرا نماید. برای ایجاد یک برنامه ی دسکتاپ ویندوز (Desktop Apllication) که از امکانات مدیر شبیه ساز استفاده نماید مراحل زیر را انجام دهید:

۱. در محیط ویژوال استودیو ۲۰۰۸ یک برنامه ی Visual C# Windows application ایجاد نمایید.
۲. یک مرجع اسمبلی Microsoft.DeviceEmulatorManager.Interop.9.0 را به پروژه اضافه کنید.

نکته: حتی اگر نگارش جدیدتری از شبیه ساز دستگاه را استفاده نمایید لازم است که فایل اسمبلی مورد نیاز در مسیر *\Program Files\Microsoft Device Emulator\1.0\Microsoft.DeviceEmulatorManager.Interop.9.0.dll* موجود باشد چرا که نگارش جدیدتر شبیه ساز دستگاه در اصل یک ارتقا از نگارش پیشین آن است.

۳. برای نمایش تمامی اطلاعات مربوط به شبیه ساز های ذخیره شده در کامپیوتر خود، به صورت یک نمودار درختی (TreeView Control) از قطعه کد زیر در بخش مدیر رویداد Load فرم اصلی برنامه تان استفاده نمایید. این کد از اینترفیس IDeviceEmulatorManager* استفاده می کند.

(* ر.ک http://msdn.microsoft.com/en-us/library/bb531169.aspx)

```
string categoryName;
TreeNode categoryNode;
IDeviceEmulatorManager emulatorManager = new DeviceEmulatorManagerClass();

emulatorManager.Reset();
try
{
    while (true)
    {
         categoryName = emulatorManager.get_Name();
         categoryNode = new TreeNode(categoryName);
         ListDeviceEmulatorSDKs(emulatorManager, categoryNode);
         tvEmulators.Nodes.Add(categoryNode);
         emulatorManager.MoveNext();
    }
}
catch (COMException ex)
{
    if (ex.ErrorCode != END_OF_DATA)
         throw ex;
}
```

۴. به روش مشابه می توانید اطلاعات تمامی شبیه ساز های نصب شده را با استفاده حلقه ی while و اینترفیس IEnumManagerSDKs** به دست آورید.

(** ر.ک http://msdn.microsoft.com/en-us/library/bb531190.aspx)

۵. پس از بازیابی اطلاعات تمامی خانواده ی شبیه ساز ها، با روش مشابه خواهید توانست اطلاعات تک تک شبیه ساز های متعلق به هر SDK را در حلقه ی while دیگری با استفاده از اینترفیس IEnumVMIDs *** به دست آورید.

(*** ر.ک http://msdn.microsoft.com/en-us/library/bb531183.aspx)

۶. پس از دسترسی به یک شبیه ساز خاص، با استفاده از روال ها می توانید آن را کنترل نمایید. مثلا با فراخوانی روال هایConnect ، Cradle، Uncradle و Shutdown در اینترفیس IDeviceEmulatorManagerVMID ****.

(**** ر.ک http://msdn.microsoft.com/en-us/library/bb531187.aspx)

فصل سوم

توسعه ی برنامه با WinForm

مقدمه

توسعه ی نرم افزار ویندوز موبایل شباهت زیادی به توسعه ی نرم افزار در دسکتاپ دارد به ویژه زمانی که یکی از دو زبان ویژوال بیسیک یا ویـژوال سی شارپ دات نت را استفاده می کنید. شما همان ابزارهای توسعه ی برنامه های ویندوز دسکتاپ را برای ویندوز موبایل هـم اسـتفاده مـی کنیـد لیکن تفاوت هایی نیز بین این دو محیط هست. دستگاه های ویندوز موبایل صفحه ی نمایش کوچک تر و منابع محدود تر دارند و همچنـین قابـل حمل بوده، اغلب باتری استفاده می کنند.

این مقاله اطلاعاتی را در باره ی چگونگی نرم افزارهای ویندوز فرم برای دستگاه های ویندوز موبایل به شما خواهـد داد و در نهایـت شـما یـاد مـی گیرید که چطور نرم افزارها را برای استفاده از امکانات دستگاه های ویندوز موبایل توسعه دهید و در عین حال حواس تان به شارژ باتری هم باشـد و صفحه نمایش برنامه تان بدون توجه به جهت صفحه ی نمایش دستگاه، درست عمل کند.

دستگاه های ویندوز موبایل مدل های متنوعی دارند

دستگاه های ویندوز موبایل در بازار بسیار متنوع هستند و به علاوه نگارش های مختلفی از ویندوز موبایل را استفاده می کنند. در این مقالـه مـا بـا نگارش ۶ ویندوز موبایل (هر دو ویرایش استاندارد و حرفه ای) کار خواهیم کرد.

دستگاه های ویندوز موبایل استاندارد

این دستگاه ها اغلب به نام اسمارت فون (تلفن هوشمند) در بازار هستند. همه ی این دستگاه ها امکانـات تلفـن را دارنـد لـیکن صـفحه ی لمسـی (Touch Screen) ندارند. معمولا برای کار با آن ها به یک دست نیاز دارید (one-hand operation). برای توسعه نرم افزار هایی کـه بـا ایـن دستگاه ها کار کنند شما ابزار توسعه ی ویندوز موبایل ۶ استاندارد (Windows Mobile 6 Standard SDK) را همراه بـا ویـژوال اسـتودیو ۲۰۰۵ یا بالاتر نیاز دارید.

دستگاه های ویندوز موبایل ۶ حرفه ای

این دستگاه ها عمدتا به نام پاکت پی سی (Pocket PC=PPC) شناخته می شوند. این دستگاه ها بعضا امکانات تلفن را نیز دارند ولـی همـه ی آن ها به صفحه ی لمسی مجهز هستند. معمولا برای کار با این دستگاه ها به دو دست همراه با یک کنترل لمسی نیاز دارید. (. بـرای توسـعه نـرم افزار هایی که با این دستگاه ها کار کنند شما ابزار توسعه ی ویـنـدوز موبایـل ۶ حرفـه ای (Windows Mobile 6 Professional SDK) را همراه با ویژوال استودیو ۲۰۰۵ یا بالاتر نیاز دارید.

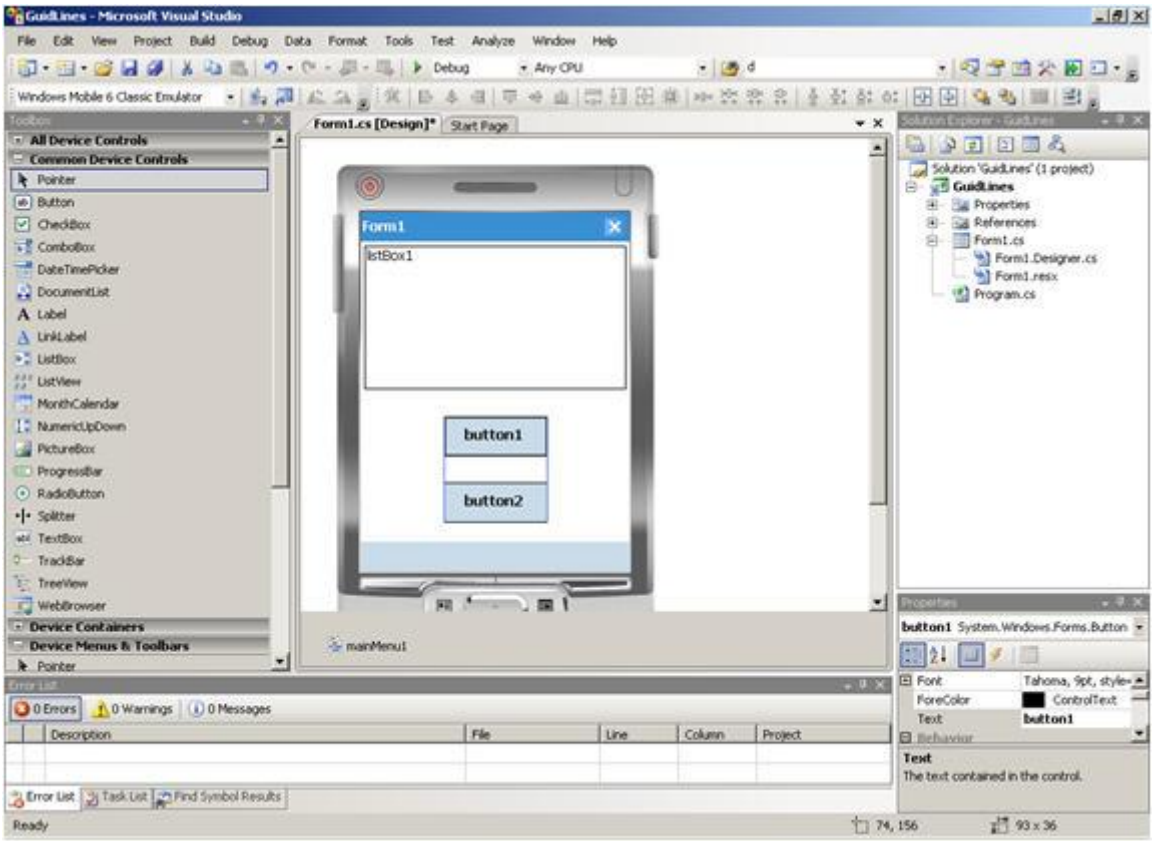
مهارت هایی که توسعه دهنده باید داشته باشد

روش های توسعه ی نرم افزار های ویندوز موبایل و دسکتاپ در محیط ویژوال استودیو ۲۰۰۸ تفاوت چندانی با هم ندارند. پس از ورود بـه محـیط ویژوال استودیو ۲۰۰۸ شما تصمیم می گیرید که چه نوع پروژه ای را و در چه پلاتفرمی آغاز کنید. سپس ویژوال استودیو یـک پـروژه ی اولیـه بـا یک فرم خالی نمایش می دهد که در آن فرم، واسط کاربر را طراحی کنید. عمده ترین تفاوت به هنگام توسعه ی نرم افزار ویـنـدوز موبایـل، صـفحه نمایش کوچک و در عین حال متنوع مورد استفاده در دستگاه ها است. تفاوت بعدی را زمانی متوجه می شوید که که بخواهید کنترل هـای واسـط کاربر را از جعبه ابزار ویژوال استودیو ۲۰۰۸ بکشید و روی فرم قرار دهید. تعداد محدودی کنترل در مقایسه با پلاتفرم دسکتاپ در دسـترس شـما خواهد بود. این محدودیت در ویرایش استاندارد بیشتر از ویرایش حرفه ای ویندوز موبایل است. از آن جا که دستگاه های ویندوز موبایل اسـتاندارد صفحه ی لمسی ندارند، برخی کنترل های ویرایش حرفه ای ویندوز موبایل در ویرایش استاندارد کارآیی ندارند.

ایجاد واسط کاربری برای برنامه یعنی همان کشیدن کنترل ها از جعبه ابزار ویژوال استودیو و رها کردن و قرار دادن آن روی یک فرم یا داخل یک کنترل کانتینر (Container) - که برای گروه بندی کنترل ها روی فرم قرار داده اید- و از طرفی می توانید چند فرم در برنامه تان داشته باشـید، خودتان کنترلی درست کنید که از یک کنترل موجود مشتق شده باشد یا این که از ابتدا یک کنترل را کاملا خودتان ایجاد کنیـد. عـلاوه بـر ایـن کنترل های ساده تعدادی پنجره ی مکالمه هم هست که شما را کمک خواهد کرد تا برنامه تان به برنامه های عمومی و اسـتاندارد ویـنـدوز موبایـل شبیه باشد.

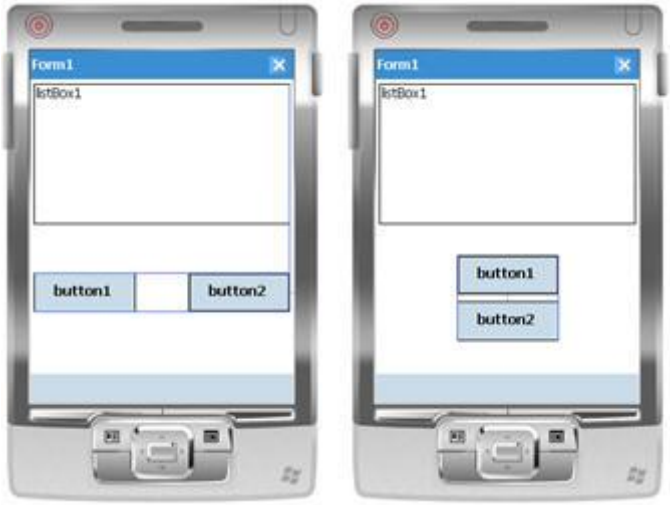
طراحی یک واسط کاربر ساده در ویندوز موبایل ۶ حرفه ای

یک پروژه ی جدید در محیط ویژوال استودیو ۲۰۰۸ ایجاد کنید. نوع پروژه باید Smart Device project در یکی از زبان هـای C# یـا ویـژوال بیسیک دات نت باشد. پلاتفرم مقصد را Windows Mobile 6 Professional SDK انتخاب نمایید. ویژوال استودیو ۲۰۰۸ یک فرم خالی با ابعاد متناسب با پلاتفرم مقصد به شما نمایش خواهد داد که به شکل یک دستگاه ویندوز موبایل است تا به حد امکان شبیه تر باشد بـه آن چـه در واقعیت استفاده می گردد. با استفاده از جعبه ابزار Device Controls (شکل ۱) می توانید کنترل های مورد نظر تان را روی فرم قرار دهید.



شکل ۱۲: ویژوال استودیو ۲۰۰۸ در نمای طراحی فرم

خب! بیکار نمانید و کنترل های مختلفی را که می توانید در طراحی واسط کاربر استفاده کنید، نگاهی بیندازید. اگر برخی از آن ها بـه چشـم تـان آشنا نیستند بهترین کار این است که آن ها روی فرم قرار دهید و مشخصه های آن ها و کارکرد هر کدام را مطالعـه کنیـد و صـد البتـه راهنمـای آنلاین را هم به یاد داشته باشید. یکی از مهم ترین چیزهایی که به هنگام طراحی واسط کاربر به چشم خواهد آمد، نحوه ی کار طراح واسط کـاربر است. دقت کنید که چطور کمک تان خواهد کرد محل قرار گیری کنترل ها را تراز کنید و کمترین فاصله ی ممکن را برای آن ها در نظر بگیریـد. خطوط راهنما (Guide Lines) کمک زیادی خواهند کرد تا در اندک زمانی، واسط کاربر را خیلی خوش دست طراحی کنیـد. در شـکل ۲ قـرار گیری دکمه های button1 و button2 را در دو حالت افقی و عمودی نسبت به هم می بینید که به کمک خطوط باریک آبی رنگ، لبه هـای آن ها نسبت به هم تراز شده اند.



شکل ۱۳: خطوط راهنما بهترین دستیار شما در ترکیب بندی واسط کاربر

ایجاد نرم افزاری که بتواند با جهت های مختلف صفحه نمایش سازگار باشد

تنوع زیاد دستگاه ها و مدل های آن ها باعث شده است که توسعه ی یک نرم افزار که بتواند در تعداد متنوعی دستگاه کار کند اندکی پیچیده باشد. ویژوال استودیو به شما امکان بندکشی و ثابت کردن کنترل های واسط کاربر را فراهم می کند. با ابزار توسعه ی ویندوز موبایل ۶ می توانید اطلاعاتی در باره ی قابلیت های دستگاه و نگارش های سیستم عامل دستگاه به دست آورید. لیکن در این مقاله ما روی چند مورد خاص و محدود کار می کنیم:

- توجه به میزان منابع آزاد سیستم و باتری
- اجرا در دستگاه های ویندوز موبایل ۶
- نمایش درست در هر دو حالت افقی، عمودی و همچنین دستگاه هایی با صفحه نمایش مربع شکل

با وجودی که (در زمان نگارش این مقاله) دستگاه های ویندوز موبایل استاندارد جهت های مختلف صفحه ی نمایش را پشتیبانی نمی کنند ولی ممکن است در آینده این امکان فراهم شود. در برخی موارد نمایش اطلاعات در صفحه ی افقی باعث راحتی بیشتر کاربر خواهد شد. ویژوال استودیو به هنگام طراحی واسط کاربر یک برنامه ی ویندوز فرم، کمک تان خواهد کرد تا کنترل ها در یک موقعیت خاص ثابت کرده، یا بندکشی کنید.

بندکشی یا ثابت کردن کنترل ها

برای مطمئن شدن از این که یک فرم در هر دو حالت افقی و عمودی به درستی نمایش داده خواهد شد، می توانید کنترل ها را در محل خاصی داخل یک کنترل مافوق (Parent) مانند فرم یا پانل (Panel Control) ثابت یا بندکشی نمایید.

نکته ی مهم تفاوت بین بندکشی (Dock) و تثبیت (Anchor) کنترل ها است. زمانی که شما یک کنترل را نسبت به یک یا چند لبه ی یک کانتینر ثابت می کنید موجب خواهد شد که مختصات نسب به لبه های ثابت شده با تغییر جهت نمایش ثابت بماند. هنگام بندکشی یک کنترل شما لبه ای از کانتینتر را که کنترل باید در آن موقعیت قرار بگیرد، تعیین می کنید. اگر کنترل را به هر چهار لبه بندکشی کنید در این صورت آن کنترل تمام سطح کنترل مافوقش (فرم یا کانتینر) را خواهد پوشاند.

برای فهمیدن بهتر تفاوت بندکشی و تثبیت کنترل ها و کارکرد آن ها در جهت های مختلف صفحه ی نمایش یک فرم ساده ایجاد می کنیم و تعدادی کنترل برچسب را (Label) - که هر کدام رنگ پس زمینه ی متفاوت با بقیه داشته باشد – روی آن قرار می دهیم. البته این واسط کاربری شاید به نظر بی مصرف باشد لیکن مهم ترین استفاده ی آن در حال حاضر همین فهمیدن بندکشی و تثبیت کنترل ها و تغییر وضعیت آن ها به هنگام تغییر جهت بین افقی و عمودی است. (شکل ۳)

شکل ۱۴: ویندوز موبایل ۶ حرفه ای در حالت عمودی، کنترل ها نه ثابت شده اند و نه بندکشی

سه برچسب بالایی روی یک پانل قرار گرفته اند و دو برچسب پایینی روی خود فرم. برای تمامی کنترل هایی که روی فرم قرار می دهید باید مشخصه های تثبیت آن را حذف کنید چرا که ویژوال استودیو به صورت پیش فرض همه ی کنترل ها را نسبت به لبه های بالا و سمت چپ کانتینر شان ثابت می کند. واسط کاربر در حالت عمودی خوب به نظر می آید ولی به محض تغییر جهت از عمودی به افقی مشکلاتی پیش می آید. کافی است به شکل ۴ توجه کنید.

شکل ۱۵: ویندوز موبایل ۶ حرفه ای در حالت افقی، کنترل ها نه ثابت شده اند و نه بندکشی

بلافاصله خواهید دید که تمامی کنترل های برچسب که در نمایش عمودی قابل دیدن بودند در نمایش افقی دیده نمی شوند. بخشی از پانلی که شامل برچسب بالایی است، از صفحه بیرون رفته است. به همین حالت برچسب پایینی کاملا محو شده است و سمت راست و چپ صفحه ی نمایش در حالت عمودی خالی مانده اند. این یعنی حالت افقی صرفا بخشی از صفحه نمایش حالت عمودی واسط کاربر را نشان می دهد.

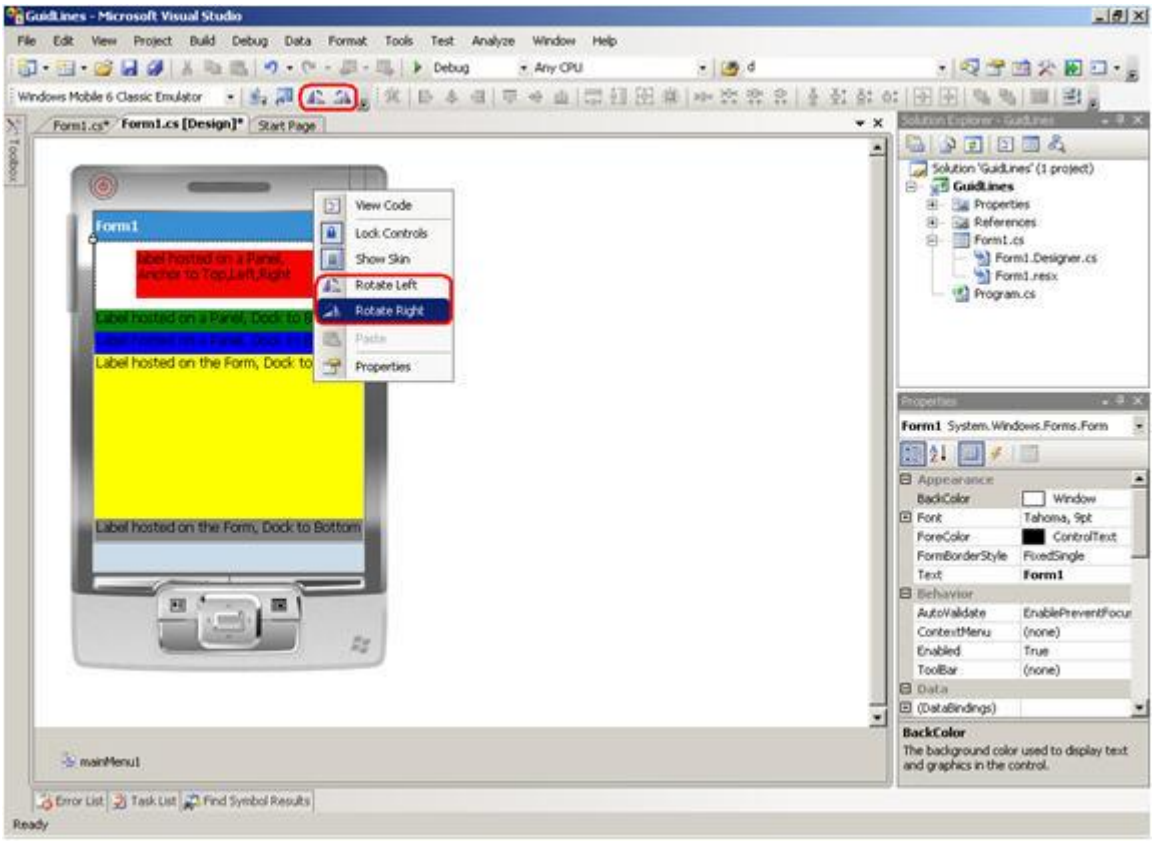
برای رفع این مشکل بدون نیاز به کدنویسی کافی است از بندکشی و تثبیت کنرل ها به درستی استفاده نمایید. با استفاده از مشخصه ی Anchor می توانید یک کنترل را نسبت به یک یا چند لبه ی صفحه ی نمایش، ثابت نگه دارید. یعنی مختصات آن نسبت به لبه های تعیین شده در هر حالتی از جهت نمایش ثابت می ماند. با استفاده از مشخصه ی Dock نیز تعیین می کنید که یک کنترل چگونه نسبت به لبه های کنترل مافوق خودش تراز شده، یا بخشی از صفحه ی نمایش را پر کند. با بندکشی چند کنترل به یک لبه کنترل مافوق شان، باعث خواهد شد

کنترل های مورد نظر به صورت پشت سر هم قرار گیرند. در این مثال سه برچسب بالایی داخل یک پانل قرار گرفته اند. بالاترین برچسب بندکشی نشده است و نسبت به لبه های چپ، راست و بالای پانل ثابت شده است. به عبارتی می توان گفت که به سمت بالای فرم بندکشی شده است. در هر دو حالت عمودی و افقی فاصله ی برچسب با لبه های بالا، چپ و راست تغییر نخواهد کرد. به طور مشخص در حالت نمایش افقی اندازه برچسب بزرگ تر می شود تا فاصله های مذکور تغییر نکنند. دو بر چسب دیگر داخل پانل – که روی هم نمایش داده می شوند- به سمت پایین پانل بندکشی شده اند. شماره ی ترتیب (tab order) مشخص می کند که کدام یک از برچسب ها نمایش داده شوند. اگر جهت نمایش را به افقی تغییر دهید، همه ی کنترل های روی فرم همچنان قابل مشاهده خواهند بود و همه ی سطح فرم توسط کنترل ها پر خواهد شد. (شکل ۵)

شکل ۱۶: ویندوز موبایل ۶ حرفه ای در حالت افقی، با کنترل های ثابت و بندکشی شده

واسط کاربری که در شکل ۵ می بینید، به مراتب بهتر از شکل ۴ است، چون کنترل ها برای نمایش افقی تغییر اندازه داده شده اند. برای این که دست تان بیاید چطور با استفاده از تثبیت یا بندکشی، کنترل ها را جا نمایی کنید کافی است کمی بیشتر وقت گذاشته، حالت های مختلف را آزمایش کنید.

در محیط طراحی ویژوال استودیو ۲۰۰۸ به راحتی هر دو حالت عمودی و افقی را می توانید آزمایش کنید. روی شکل دستگاه راست کلیک و گزینه ی Rotate Left یا Rotate Right را انتخاب نمایید. (شکل ۶) راه دیگر آن است که فرم را انتخاب (کلیک) کرده، دکمه های روی نوار ابزار را برای تغییر جهت نمایش استفاده کنید.



شکل ۱۷: تغییر جهت نمایش در حالت طراحی

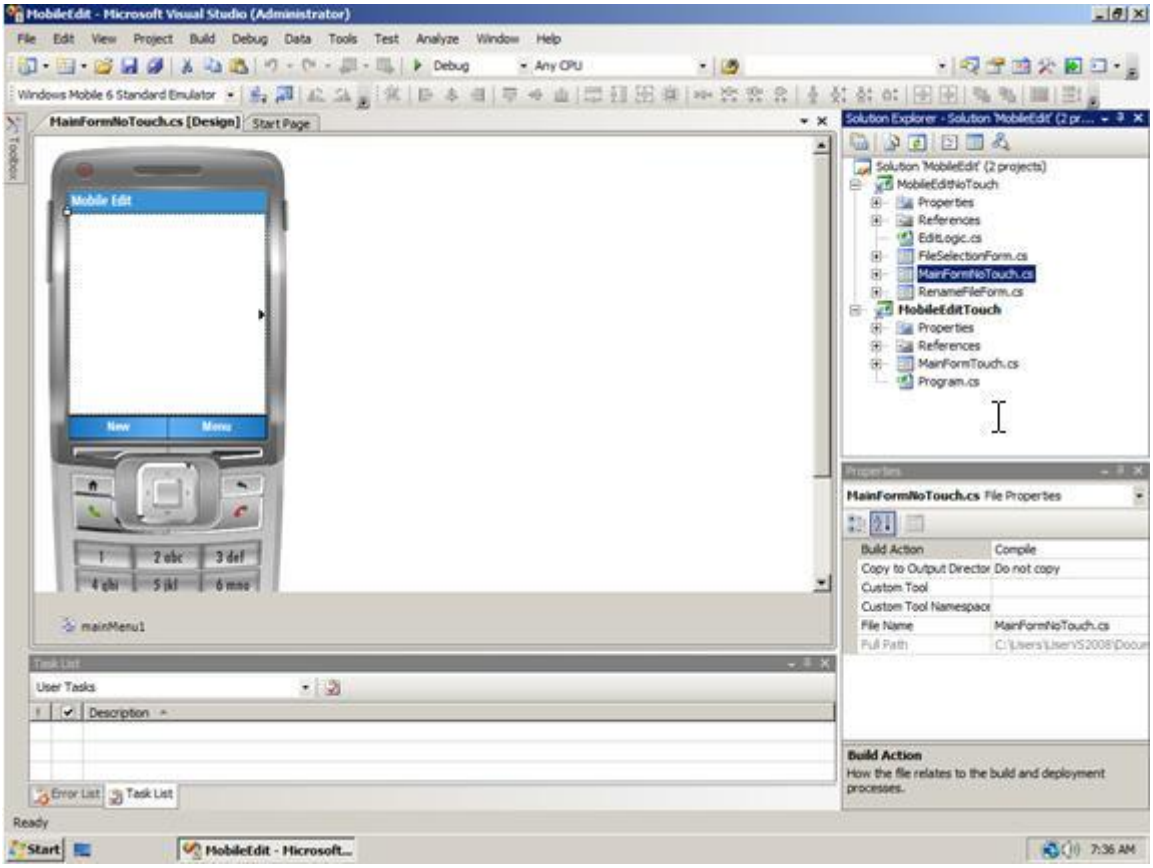
طراحی برنامه ای که روی دستگاه های مختلف کار کند

به هنگام ایجاد نرم افزاری که روی یکی از پلاتفرم های مقصد ابزار توسعه ی ویندوز موبایل ۶ (اعم از استاندارد یا حرفه ای) کار کند، می توانید پلاتفرم فعلی را به پلاتفرم مقصد در ابزار توسعه ی دیگری تغییر دهید. در این حالت نرم افزار به پلاتفرم دیگری تبدیل می شود یعنی ویژوال استودیو ۲۰۰۸ برنامه ی شما را تغییر می دهد. اگر بخواهید برنامه ای را که در ویندوز موبایل ۶ حرفه ای ساخته اید، به پلاتفرم دیگری تبدیل کنید مشکلاتی را پیش روی خود خواهید داشت. از آن جا که ویرایش استاندارد ویندوز موبایل ۶ به نسبت ویرایش حرفه ای آن کنترل های کمتری را در واسط کاربر پشتیبانی می کند، کنترل های پشتیبانی نشده را باید به کنترل هایی تغییر دهید که در پلاتفرم جدید کامپایل و اجرا شوند. ویژوال استودیو ۲۰۰۸ کار تبدیل را به راحتی انجام خواهد داد. از دیدگاه توسعه ی نرم افزار ایده ی برتر آن است که برای ساده تر شدن بحث سازگاری پلاتفرم ها، لایه ی بیزینس لاجیک* را از لایه ی واسط کاربر جدا کنیم. کنترل های واسط کاربر که در پلاتفرم جدید پشتیبانی نمی شوند، توسط ویژوال استودیو ۲۰۰۸ علامت گذاری می شوند و در عین حال ویژوال استودیو ۲۰۰۸ به شما پیشنهاد می دهد که کنترل های مذکور را به نزدیک ترین حالت منطبق با پلاتفرم جدید تبدیل کنید. اگر نخواستید که پروژه تان را به پلاتفرم مقصد دیگری تبدیل کنید می توانید از امکانات ارث بری فرم ها (Visual Form Inheritance) استفاده کنید تا با یک دفعه کدنویسی بتوانید برنامه تان را در پلاتفرم های مختلف اجرا نمایید.

* Business Logic این اصطلاح را ترجمه نکردم. اگر اهل فن هستید تصدیق می کنید که نباید این چنین عبارتی را ترجمه کرد. اگر هم این اصطلاح برای تان آشنا نیست، **«به گیرنده های خود دست نزنیدا»** کافی است مطالبی در باره ی معماری لایه به لایه ی نرم افزار را مطالعه کنید. (مترجم)

ارث بری فرم ها

ویژوال استودیو ۲۰۰۸ امکانات کامل طراحی را در زمان ارث بری یک فرم از فرم دیگر، به شما می دهد. برای فهمیدن بهتر، با یک فرم پایه شروع می کنیم که نقطه ی آغازی خواهد بود برای تولید نرم افزاری که در پلاتفرم های مختلف ویندوز موبایل کار کند. با وجود این که ممکن است این کار در نگاه اول غیر ضروری به چشم بیاید، لیکن تجربه ی ایجاد یک فایل کتابخانه ای کلاس (class library) - که فرم پایه را ایجاد می کند- بسیار ارزش مند خواهد بود. این موضوع زمانی که شما یاد بگیرید کدام کنترل ها در فرم های ارث بری شده قابل تغییر هستند و کدام شان نیستند، اهمیت بیشتری دارد. برای این که بتوانید کنترل ها یا کارکرد آن ها را تغییر دهید باید مشخصه ی Access Modifier کنترل مورد نظر را به مقداری غیر از Private تغییر دهید. در شکل ۷ یک فرم پایه می بینید که به عنوان مرحله ی آغاز برنامه تان استفاده می شود.

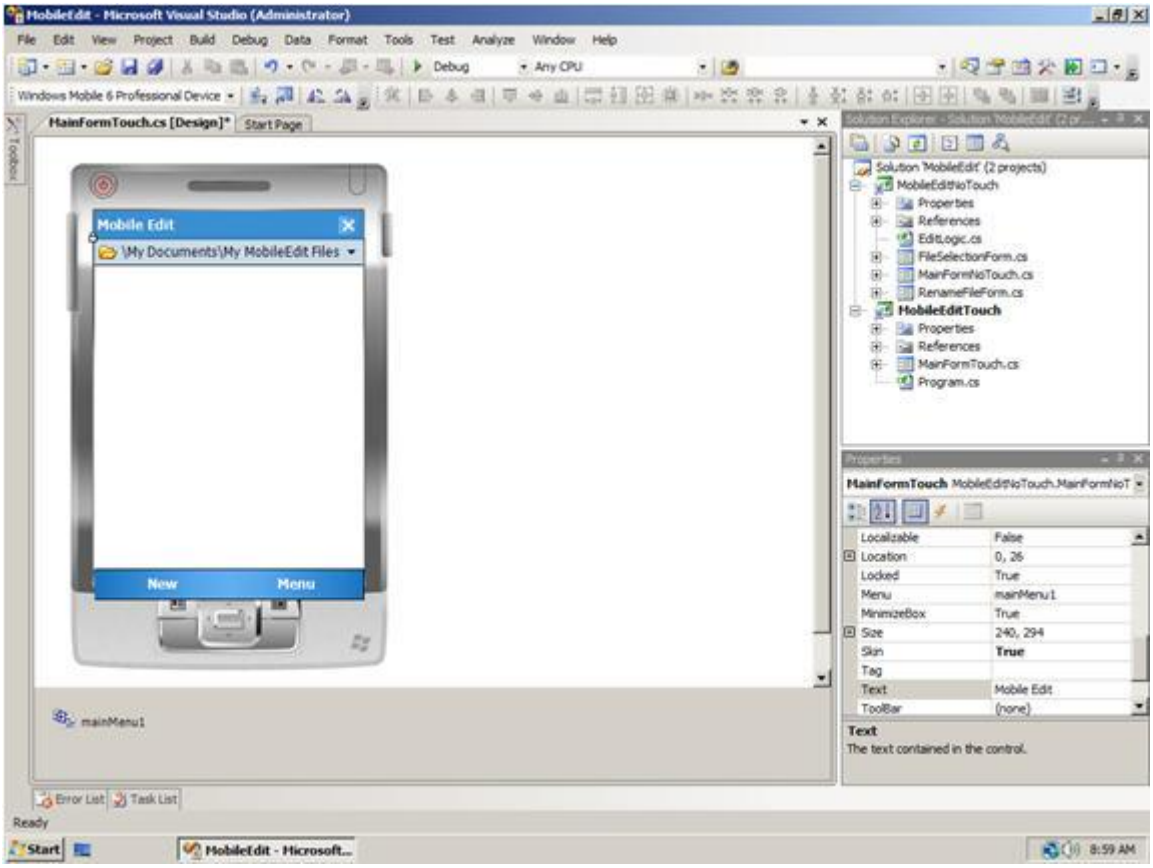


شکل ۱۸: فرم پایه ی برنامه برای دستگاه ویندوز موبایل استاندارد

فرم پایه حاوی یک کنترل ویرایشگر (Edit) - که همه ی صفحه را پر کرده- و یک منو است. این برنامه یک ویرایشگر متن ساده است. این برنامه فایل متنی ایجاد می کند، محتوای فایل متنی موجود را تغییر می دهد و همچنین امکان تغییر نام یا حذف فایل متنی را فراهم می کند. با این که فایل های ایجاد شده توسط این برنامه همان فایل های متنی عادی هستند، لیکن برای مدیریت راحت آن ها و فیلتر کردن شان، پسوند فایل را مخصوص این برنامه قرار داده ایم. فرم پایه برای ویندوز موبایل استاندارد طراحی شده است. دلیلش هم روشن است. ویندوز موبایل استاندارد نسبت به حرفه ای کنترل های کمتری دارد. برای مدیریت فایل ها هم دو فرم دیگر در نظر گرفته ایم که یکی فهرست فایل های موجود را نشان دهد و دیگری برای تغییر نام یا حذف آن ها استفاده شود. از آن جا که ویندوز موبایل استاندارد، صفحه ی لمسی را پشتیبانی نمی کند تمامی عملیات فایل ها از طریق گزینه های منو انجام می شوند. در این مقاله ما توضیحات مربوط به ویرایشگر متنی را نخواهیم گفت و روی ایجاد نرم افزاری که روی انواع مختلف دستگاه ها کار کند متمرکز می شویم. هر چند که برنامه ی نمونه ی Mobile Edit برای دانلود دم دست مان هست.

در شکل ۸ شما فرم اصلی را در حال اجرا روی ویندوز موبایل ۶ حرفه ای می بینید. با وجود تفاوت ظاهری اش با آن چه در شکل ۷ دیده اید، ولی این فرم از همان فرم اصلی که در شکل ۷ دیدید مشتق شده است. فقط یک کنترل مخصوص ویندوز موبایل ۶ حرفه ای به آن اضافه شده است. یعنی یک Document List و همان طور که از نامش پیدا است برای نمایش فهرست فایل ها به کار می رود. با این که فرم اصلی حاوی یک

کنترل ویرایشگر متن است ولی به هنگام طراحی آن را نمی بینید چون کنترل «فهرست فایل ها» (Document List) روی آن قرار گرفته است. داخل برنامه با کدنویسی کاری می کنیم که کنترل «فهرست فایل ها» در زمانی که لازم نیست نشان داده نشود. کنترل هایی که بخشی از فرم پایه هستند در فرم مشتق شده هم به ارث می رسند. کنترل هایی را که از طریق ارث بری در فرم جدید قرار گرفته اند، با یک علامت فلش در گوشه ی بالا سمت چپ شان (همانند کنترل منو) مشخص می شوند.



شکل ۱۹: فرم پایه ی مشتق شده برای ویندوز موبایل حرفه ای

از آن جا که دستگاه های ویندوز موبایل حرفه ای از کنترل های بیشتری پشتیبانی می کنند، خود نرم افزار ساده تر می شود و به راحتی از امکان صفه ی لمسی استفاده خواهد کرد. امکان استفاده از کنرل «فهرست فایل ها» برای مدیریت فایل ها یعنی این که دیگر نیازی به طراحی فرم جدید نیست. از آن جا که هدف اصلی ما ایجا یک فایل باینری (اجرایی) برای استفاده در هر دو پلاتفرم ویندوز موبایل استاندارد و حرفه ای است، به هنگام شروع اجرای آن می توانیم دستگاهی را که برنامه در آن اجرا می شود، تشخیص دهیم و فرم مناسب آن را نمایش دهیم.

.NET Compact Framework 3.5 به راحتی امکان تشخیص نوع دستگاه را برای تان فراهم می کند. قطعه کد زیر را - که بخشی از فایل Program.cs است – ببینید:

```
static void Main()
{
    if (SystemSettings.Platform == WinCEPlatform.PocketPC)
    {
        Application.Run(new MainFormTouch());
    }
    else if (SystemSettings.Platform == WinCEPlatform.Smartphone)
    {
        Application.Run(new MainFormNoTouch());
    }
}
```

بسته به نوع دستگاه، یکی از دو فرم MainFormNoTouch یا MainFormTouch را به عنوان فرم اصلی برنامه مان استفاده می کنیم. مزیت اصلای ارث بری فرم این است که ما امکانات فرم اصلی (Main) را در ویندوز موبایل حرفه ای، دوباره استفاده می کنیم. مثلا اگر کاربر گزینه ی New را در منو کلیک کند، صرف نظر از نوع دستگاهی که برنامه در آن اجرا می شود، مدیر رویدادی (event handler) که برای گزینه ی New ی منو در صفحه ی MainFormNoTouch تعریف کرده ایم، اجرا خواهد شد:

```
 protected virtual void menuNew_Click(object sender, EventArgs e)
{
     autoSave = false;
     if (tbWorkArea.Text.Length != 0)
     {
           if (MessageBox.Show("Save contents before creating a new file",
               "Save current contents",
               MessageBoxButtons.YesNo,
               MessageBoxIcon.Question,
               MessageBoxDefaultButton.Button1) == DialogResult.Yes)
           {
               EditLogic.WriteFile(fileName, tbWorkArea.Text);
           }
           tbWorkArea.Text = string.Empty;
      }
      autoSave = true;
      fileName = EditLogic.FileName;
}
```

این کد از کاربر خواهد پرسید که آیا فایل های در حال استفاده را قبل از ایجاد فایل جدید، ذخیره کند یا نه. همان طور که می بینید امکانات خود ویرایشگر در کلاس دیگری به نام EditLogic قرار دارد که لایه ی واسط کاربر را به طور کامل از لایه ی اپلیکیشن لاجیک (Application Logic) جدا می کند. اگر برنامه در دستگاه با سیستم عامل ویندوز موبایل حرفه ای اجرا شود، یکی از قابلیت هایی که باید در ابتدای اجرای برنامه غیر فعال شود کنترل «فهرست فایل ها» است. این کار را با ابطال مدیر رویداد menuNew_Click مطابق قطعه کد زیر انجام می دهیم:

```
protected override void menuNew_Click(object sender, EventArgs e)
{
     base.menuNew_Click(sender, e);
     documentList.Visible = false;
     MinimizeBox = false;
}
```

زمانی که کاربر در ویندوز موبایل حرفه ای گزینه ی New را انتخاب می نماید، ابتدا کد نوشته شده در فرم پایه اجرا می شود سپس دو خط کد که فقط در ویندوز حرفه ای در دسترس هستند اجرا می شوند. این دستور ها به ترتیب برای مخفی کردن کنترل «فهرست فایل ها» برای دیده شدن کنترل «جعبه متن» و نمایش یک دکمه ی OK در نوار عنوان فرم، اجرا می شوند.
برای ابطال مدیر رویداد menuNew_Click باید آن را به طور دستی به فایل MainFormTouch.cs اضافه نمایید. اگر روی گزینه ی New در محیط طراحی فرم دابل کلیک کنید، ویژوال استودیو یک مدیر رویداد جدید به نام menuNew_Click_1 ایجاد می کند که در زمان اجرا، پس از فراخوانی مدیر رویداد menuNew_Click اصلی در فرم پایه، فراخوانی خواهد شد.

توجه به منابع سیستم

صرف نظر از مدل دستگاه هر زمان که فرم اصلی برنامه مخفی شود (در پس زمینه قرار گیرد) یک رویداد فراخوانی می شود. به طور مشابه زمانی که فرم اصلی برنامه مجدد فعال می شود و در پیش زمینه ویندوز قرار می گیرد، رویداد دیگری فراخوانی می شود. این رویداد ها را می توانیم برای آزاد سازی منابع سیستم هنگامی که برنامه در پس زیمنه اجرا می شود، استفاده نماییم. (مثلا غیر فعال کردن تایمر) در این حال برنامه حافظه ی مورد استفاده اش را زمانی که نیازی به آن نیست، آزاد می نماید و حتی الامکان از کمترین پردازش و به تبع آن کمترین توان باتری استفاده می کند.

اگر برنامه ی Mobile Editor در پس زمینه قرار بگیرد به طور موقت محتوای جعبه متن را در یک فایل ذخیره می کند (کد زیر):

```
protected virtual void MainFormNoTouch_Deactivate(object sender, EventArgs e)
{
     if (fileName != string.Empty && tbWorkArea.Text.Length != 0)
     {
         EditLogic.WriteFile(fileName, tbWorkArea.Text);
         tbWorkArea.Text = string.Empty;
         autoSave = true;
     }
}
```

و هنگامی که برنامه دوباره در پیش زمینه قرار بگیرد، محتوای فایل مذکور را بازیابی کرده، در جعبه متن tbWorkArea نمایش می دهد (کد زیر):

```
protected virtual void MainFormNoTouch_Activated(object sender, EventArgs e)
{
     if (autoSave)
     {
         tbWorkArea.Text = EditLogic.ReadFile(fileName);
     }
}
```

تولید نرم افزار هایی که در هر دو ویرایش حرفه ای و استاندارد ویندوز موبایل کار کند، امکان پذیر است، اما این که فکری هم بـرای معمـاری نـرم افزار بکنید بسیار مهم و اساسی است. به ویژه زمانی که ملزم باشید امکانات سخت افزاری مختلف را در دستگاه های متنوع در نظر بگیرید، این کار پیچیده تر خواهد بود که صد البته از حوصله ی این مقاله خارج است.

شکل ۹: برنامه ی Mobile Edit در حال اجرا روی دستگاه های مختلف

برای استفاده از ازث بری فرم ها در محیط طراحی Windows Forms باید یک ایده ی درست و حسـابی داشـته باشـید. البتـه هـر کنترلـی را می توانید روی فرم پایه قرار دهید به شرطی که بخواهید آن را در فرم های مشتق شده نیز استفاده نمایید. فقـط یادتـان باشـد اگـر مـی خواهیـد کارکرد یک کنترل را در فرم مشتق شده تغییر دهید، باید مشخصه ی Modifiers کنترل را با protected یا public مقدار دهـی کنیـد. اگـر مشخصه ی Modifiers یک کنترل را با private مقدار بدهید، در فرم مشتق شده فقط می توانید آن را مخفی یا بندکشی کرده، یا آن را تغییـر اندازه بدهید.

فصل چهارم

افزودن کنترل سفارشی و استفاده از GPS

مقدمه

توسعه ی نرم افزار ویندوز موبایل شباهت زیادی به توسعه ی نرم افزار در دسکتاپ دارد به ویژه زمانی که یکی از دو زبان ویژوال بیسیک یا ویژوال سی شارپ دات نت را استفاده می کنید. شما همان ابزارهای توسعه ی برنامه های ویندوز دسکتاپ را برای ویندوز موبایل هم استفاده می کنید لیکن تفاوت هایی نیز بین این دو محیط هست. دستگاه های ویندوز موبایل صفحه ی نمایش کوچک تر و منابع محدود تر دارند و همچنین قابل حمل بوده، اغلب باتری استفاده می کنند.

در این مقاله یاد می گیرید که چطور نرم افزارهای بر مبنای ویندوز فرم، برای موبایل ایجاد کنید. کنترل های سفارشی را به برنامه تان اضافه کنید و چگونگی ساختن کنترل های سفارشی با پشتیبانی کامل محیط طراحی را تمرین کنید. به علاوه کاربرد سخت افزار GPS داخل نرم افزار و همچنین به روز رسانی مقادیر کنترل های واسط کاربر را داخل یک برنامه ی چند پردازشی (Multi Threaded) را نیز فرا خواهید گرفت.

افزودن کنترل های سفارشی به برنامه ی خودتان

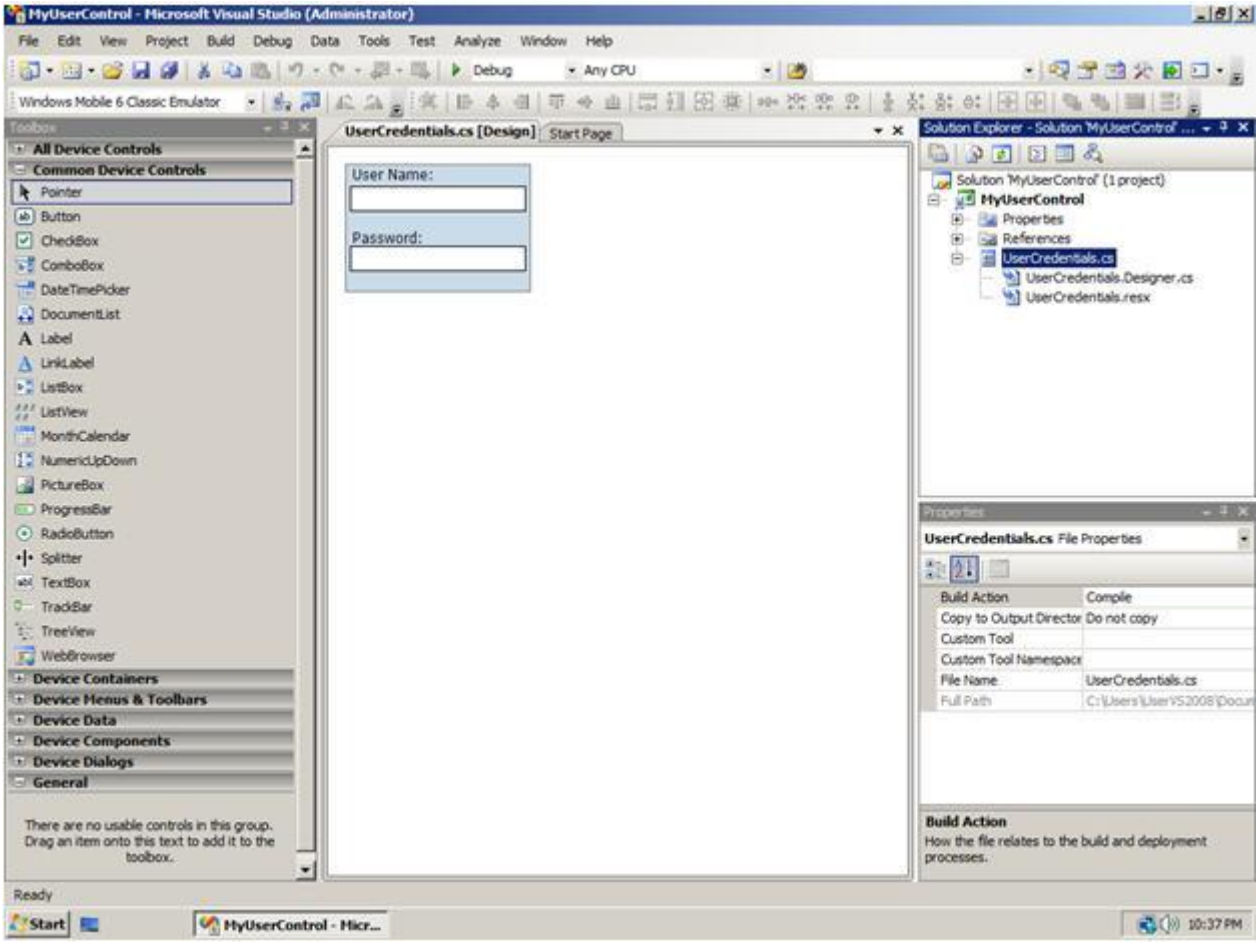
اگر به هنگام توسعه ی برنامه ی ویندوز موبایل به کنترل های خاصی نیاز پیدا کنید که در مجموعه ی کنترل های استاندارد .NET Compact Framework نیستند، راه حل های زیر را پیش رو خواهید داشت:

- یک کنترل واسط کاربر فقط برای برنامه ی فعلی تان درست کنید
- یک کنترل واسط کاربر که در برنامه ی دیگر هم قابل استفاده باشد، درست کنید
- یک کنترل واسط کاربر با امکانات مورد نظرتان خریداری کنید

در این مقاله سه روش ایجاد کنترل ها را هم از ابتدا و هم از طریق استفاده ی مجدد و ترکیب با کنترل های موجود یاد خواهید گرفت.

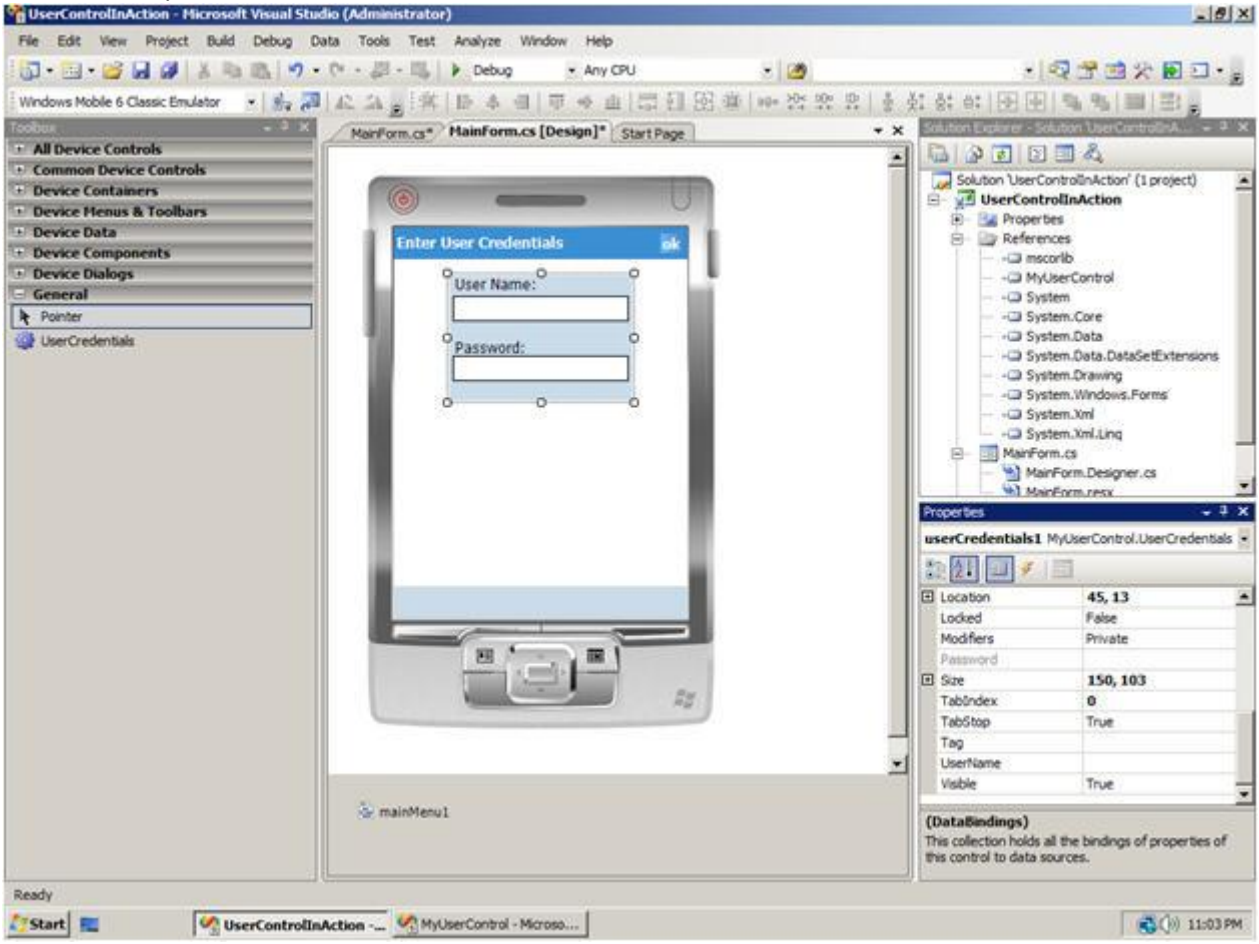
کنترل های کاربر (سفارشی)

این کنترل ها معمولا با ترکیب چند کنترل موجود و درست کردن یک کنترل جدید به کار می روند. در ویژوال استودیو ۲۰۰۸ ایجاد کنترل های کاربر به طور کامل توسط یکی از انواع پروژه ی آن (User Control Project) پشتیبانی می شود. وقتی یک کنترل جدید ایجاد می کنید یک صفحه ی طراحی مانند کانتینر پدید می آید که کنترل ها را روی آن قرار می دهید. با یک کنترل کاربر شما می توانید چند کنترل موجود را با هم ترکیب کنید به طوری که به هنگام استفاده از کنترل ترکیبی عملکرد آن مانند یک کنترل منفرد باشد. در شکل ۱ شما یک کنترل کاربر داخل ویژوال استودیو ۲۰۰۸ می بینید. کنترل مذکور شامل دو کنترل برچسب (Label) و دو کنترل جعبه متن (TextBox) است که می توانید برای اعتبار سنجی کاربر، مثلا برای ورود به یک سیستم بانک اطلاعاتی استفاده کنید. زمانی که پروژه کامپایل می شود این کنترل در اسمبلی مربوط به خودش ذخیره می شود تا در آینده توسط هر پروژه ی دیگر ویندوز موبایل استفاده گردد. در این کنترل کاربر، کدنویسی چندانی پس از قرار دادن کنترل سفارشی روی آن لازم نیست. تنها کد مورد نیاز داخل این کنترل کاربر عبارت است از تعریف دو متغیر با سطح دسترسی عمومی (Public) برای به دست آوردن نام کاربر و گذرواژه داخل یک برنامه و همچنین مقدار دهی نام کاربر است. مشخصه ی Password (گذرواژه) فقط خواندنی (Read-only) است تا کاربر را مجبور کند که حداقل یک گذرواژه ی مجاز برای خودش در نظر بگیرد. اعتبار سنجی گذرواژه در داخل برنامه انجام می شود نه داخل کنترل کاربر. دلیل این امر راحتی کار با کنترل و امکان استفاده ی مجدد از آن است.



شکل ۲۰: کنترل سفارشی برای اعتبار سنجی کاربر

برای استفاده از کنترل اعتبار سنجی کاربر داخل برنامه کافی است آن را از جعبه ابزار ویژوال استودیو بکشید و روی فرم بیندازید. کنترل اعتبرا سنجی کاربر به طور خودکار در جعبه ابزار نشان داده نمی شود. باید نخست روی بخش General جعبه ابزار راست کلیک کنید و از منوی باز شده، گزینه ی Choose Items را بزنید و به فایل اسمبلی که حاوی کنترل اعتبار سنجی کاربر است اشاره کنید تا کنترل مذکور در جعبه ابزار ظاهر شود. در شکل ۲ شما چگونگی نمایش کنترل اعتبار سنجی (User Credentals) را داخل برنامه می بینید. اگر به جعبه ابزار هم بنگرید کنترل با نام UserCredentals را در بخش General جعبه ابزار خواهید دید که به آسانی می توانید آن را روی فرم خود قرار دهید. پس از این کار شما در پنجره ی مشخصات (Properties) یک مشخصه ی UserName - که قابل مقدار دهی در محیط طراحی و در عین حال به صورت کد نویسی داخل برنامه است- می بینید. در عین حال مشخصه ی Password را با رنگ خاکستری در پنجره ی مشخصات خواهید دید که به دلیل فقط خواندنی بودن آن است. این مشخصه قابل مقداردهی نیست ولی مقدارش پس از آن که توسط کاربر به هنگام اجرای برنامه وارد شد، قابل بازیابی (خواندن) خواهد بود.



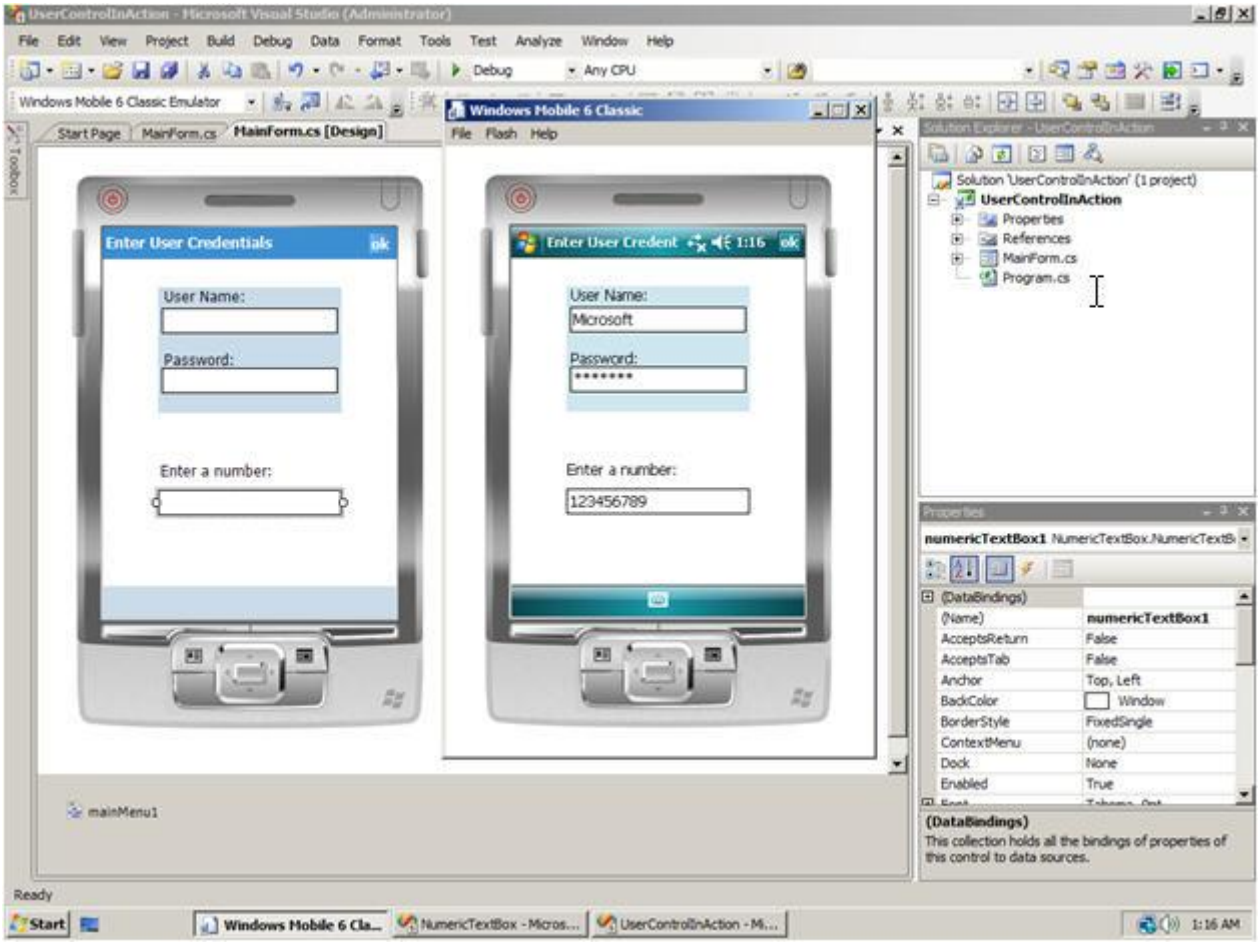
شکل ۲۱: کنترل سفارشی داخل یک برنامه

کنترل های ارث بری شده

گاهی اوقات مثلا یک کنترل عمومی .NET Compact Framework را استفاده می کنید لیکن لازم دارید که کارکرد آن مطابق دلخواه شما تغییر کند. مثلا یک جعبه متن (TextBox) که فقط عدد را بشود در آن وارد نمود. البته شما می توانید به کاربر اجازه دهید که هر چه خواست وارد نماید و پس از ورود اطلاعات، در داخل برنامه آن را اعتبار سنجی کنید. اما ایجاد یک کنترل سفارشی خواه داخل برنامه تان یا به صورت یک کنترل مجزای مشتق شده از جعبه متن عادی، که صرفا کاراکترهای عددی (دقیقا) به هنگام ورود داده ها بپذیرد ایده ی بهتری است. در این روش شما کدنویسی کمتری را در برنامه تان خواهید داشت چرا که اعتبار سنجی داخل کنترل ارث بری شده ی (inherited) جعبه متن صورت می گیرد. با وجود محدود شدن قابلیت های این جعبه متن جدید در مقایسه با جعبه متن معمولی، باز هم جعبه متن جدید امکانات کامل طراحی شامل همه ی مشخصه ها و رویداد های جعبه متن اصلی را در اختیار شما قرار می دهد. به علاوه اگر کنترل مزبور را به طور مجزا ایجاد و کامپایل کنید در برنامه های متعددی می توانید از آن استفاده نمایید. برای ایجاد یک کنترل با نام numericTextbox به صورت مجزا، نخست یک پروژه ی جدید از نوع Class Library را در برگه ی Smart Device Project انتخاب و ایجاد کنید. چند خط کد برای تبدیل یک جعبه متن معمولی به جعبه متن عددی مورد نظر، نیاز خواهید داشت که صرفا اجازه ی ورود داده های عددی و کلید BackSpace را به کاربر می دهد.

```
public class NumericTextBox : TextBox
{
     protected override void OnKeyPress(KeyPressEventArgs e)
     {
          if ((e.KeyChar < '0' || e.KeyChar > '9') && e.KeyChar != (char)Keys.Back)
          {
               SystemSounds.Beep.Play();
               e.Handled = true;
          }
          base.OnKeyPress(e);
     }
}
```

زمانی که شما کد باطل کننده ی متد اصلی را می نویسید باید تعیین کنید که متد کلاس پایه (اصلی) چه موقع می تواند فراخوانی شود. مطابق مستندات ارث بری در برنامه نویسی شیء گرا هنگامی که متد OnKeyPress را در یک کلاس مشتق شده باطل می کنیم باید مطمئن شویم که متد OnKeyPress متعلق به کلاس پایه فراخوانی می شود تا در اصل پیغام مربوط به رویداد دریافت گردد. از آنجا که کنترل کننده ی رویداد TextChanged فقط داده های عددی را دریافت می کند، شما پس از اعتبار سنجی کاراکتر های ورودی می توانید متد کلاس پایه را فراخوانی نمایید. استفاده از جعبه متن عددی داخل برنامه به همان روش استفاده از کنترل اعتبار سنجی کاربر است که در بخش قبلی همین مقاله دیدید.



شکل ۲۲: کنترل های UserCredentials (اعتبار سنجی کاربر) و NumericTextBox (ورود داده ی عددی) در حالت طراحی و اجرا داخل شبیه ساز

کنترل های سفارشی

اگر نمی خواهید کنترل های موجود را دوباره استفاده (بازنویسی) کنید ولی به کنترل های جدید نیاز دارید می توانید کنترل سفارشی خودتان را بسازید. معمولا این نوع کنترل ها از نقطه ی صفر نوشته می شوند حتی اگر بتوانید به عنوان مثال آن ها را از کلاس System.Windows.Forms.Control ارث بری نمایید که حداقل کارآیی های عمومی کنترل ها را داشته باشند. به این نوع کنترل ها دست ساز (Own Drawn Controls) هم می گویند. در ویژوال استودیو ۲۰۰۸ الگوی آماده ای برای کنترل های سفارشی نیست. آسان ترین راه همان ایجاد یک کنترل کاربر داخل ویژوال استودیو است. پس از ایجاد پروژه ی خالی می توانید کنترل کاربر را حذف کنید و به صورت دستی یک کنترل سفارشی را از ویژوال استودیو ۲۰۰۸ به آن اضافه نمایید.

```
public partial class CustomControl1 : Control
{
    public CustomControl1()
    {
        InitializeComponent();
    }

    protected override void OnPaint(PaintEventArgs pe)
    {
         // TODO: Add custom paint code here
         // Calling the base class OnPaint
         base.OnPaint(pe);
    }
}
```

برای ایجاد یک کنترل سفارشی از صفر باید خودتان آن را ترسیم (Paint) کنید. بدین منظور کافی است متد OnPaint را باطل کنید. البته این مورد از همان لحظه ی ایجاد کنترل سفارشی به طور خودکار توسط ویژوال استودیو ۲۰۰۸ به سورس برنامه تان اضافه می شود. از آ"جا که ترسیم کنترل تان به هنگام اجرای برنامه ی استفاده کننده از آن ممکن است بار ها و بار ها پیش بیاید، از این رو کارآیی روش پیاده سازی شما اهمیت دوچندان پیدا می کند. در واقع هم کارآیی و هم استفاده از منابع سیستم. از آن جا که معمولا از کلاس های مربوط به گرافیک در داخل متد OnPaint استفاده می کنید و در استفاده از این کلاس ها به کد اصلی مراجعه می کنید پاکسازی و آزاد سازی منابع مورد استفاده پس از هر بار ترسیم بسیار مهم است. در C# بهتر است از کلاس هایی استفاده کنید که یک متد Dispose را به صورت ترکیب با دستور using پیاده کرده باشند. دستور using تضمین می کند که Dispose حتی در بروز خطا به هنگام فراخوانی متدهای یک شئ هم اجرا می شود. در واقع شما از بابت مدیریت خطاها (Exception Handling) خیال تان راحت است. مثلا می خواهید یک متن را نمایش دهید.

می توانید قطعه کد زیر را استفاده نمایید و آن را داخل متد OnPaint فراخوانی کنید و شیء از نوع Graphics را به صورت بخشی از پارامتر PaintEventArgs به داخل آن بفرستید:

```
private void DisplayLabelInfo(Graphics g, string labelText, int xPos, int yPos)
{
    using (Font labelFont = new Font(FontFamily.GenericSerif, 10.0F, FontStyle.Regular))
    {
        using (SolidBrush labelBrush = new SolidBrush(LabelForeground))
        {
           g.DrawString(labelText,
                        labelFont,
                        labelBrush,
                        xPos, yPos);
        }
    }
}
```

آن چه دیدید یک متد ساده برای نمایش متن داخل یک کنترل بود. دو شیء از نوع Graphics دیگر هم لازم دارید، یکی Font و دیگری SolidBrush که هر دوی این ها به هنگام فراخوانی DisplayLabelInfo ایجاد می شوند. زمانی که این اشیا را با دستور using به کار ببرید پاکسازی آن ها کارآیی بالاتری خواهد داشت چون متد Dispose درست در زمانی که کنترل برنامه به خارج از محدوده ی تعریف (دسترسی) شیء می رود، فراخوانی می شود و در نتیجه دستورات کمتری به هنگام پاکسازی حافظه (Garbage Collection) اجرا می شود چون نیازی به فراخوانی Finalize نیست. این روش به ویژه زمانی که با کلاس های مربوط به گرافیک سر و کار دارید - به دلیل آن که آن ها از منابع محدود محلی استفاده می کنند که باید در اسرع وقت آزاد شوند- بسیار مفید است.

اگر شما مستقیما تمام ترسیم های گرافیکی را با استفاده از یک کپی نمونه ی Graphics انجام دهید صفحه نمایش حالت پرش خواهد داشت، به ویژه زمانی که کدهای زیادی را در متد OnPaint خود نوشته اید. در این حالت بهترین کار استفاده از تکنیک بافر است. یعنی این که شما هر چیزی را که می خواهید ترسیم کنید ابتدا در نمونه های جدا گانه ی Graphics در حافظه ی اصلی انجام دهید و پس از ترسیم همه ی آن ها محتوای شیء Graphics را از حافظه ی اصلی به شیء Graphics که ترسیم روی صفحه ی نمایش را انجام می دهد کپی کنید.

فرض کنید یک تصویر همراه با یک متن روی آن نمایش می دهید. برای استفاده از بافر مراحل زیر را باید انجام دهید:

```
protected override void OnPaint(PaintEventArgs pe)
{
    // Initialize a Graphics object with the bitmap we just created
    // and make sure that the bitmap is empty.
    Graphics memoryGraphics = Graphics.FromImage(memoryBitmap);
    memoryGraphics.Clear(this.BackColor);
    // Draw the control image in memory
    memoryGraphics.DrawImage(someImage,
                             destRect,
                             imageRect,
                             GraphicsUnit.Pixel);
    // Display some text
    DisplayLabelInfo(memoryGraphics, "Some text", 0, 0);
    // Draw the memory bitmap on the display
    pe.Graphics.DrawImage(memoryBitmap, 0, 0);
    // Calling the base class OnPaint
    base.OnPaint(pe);
}
```

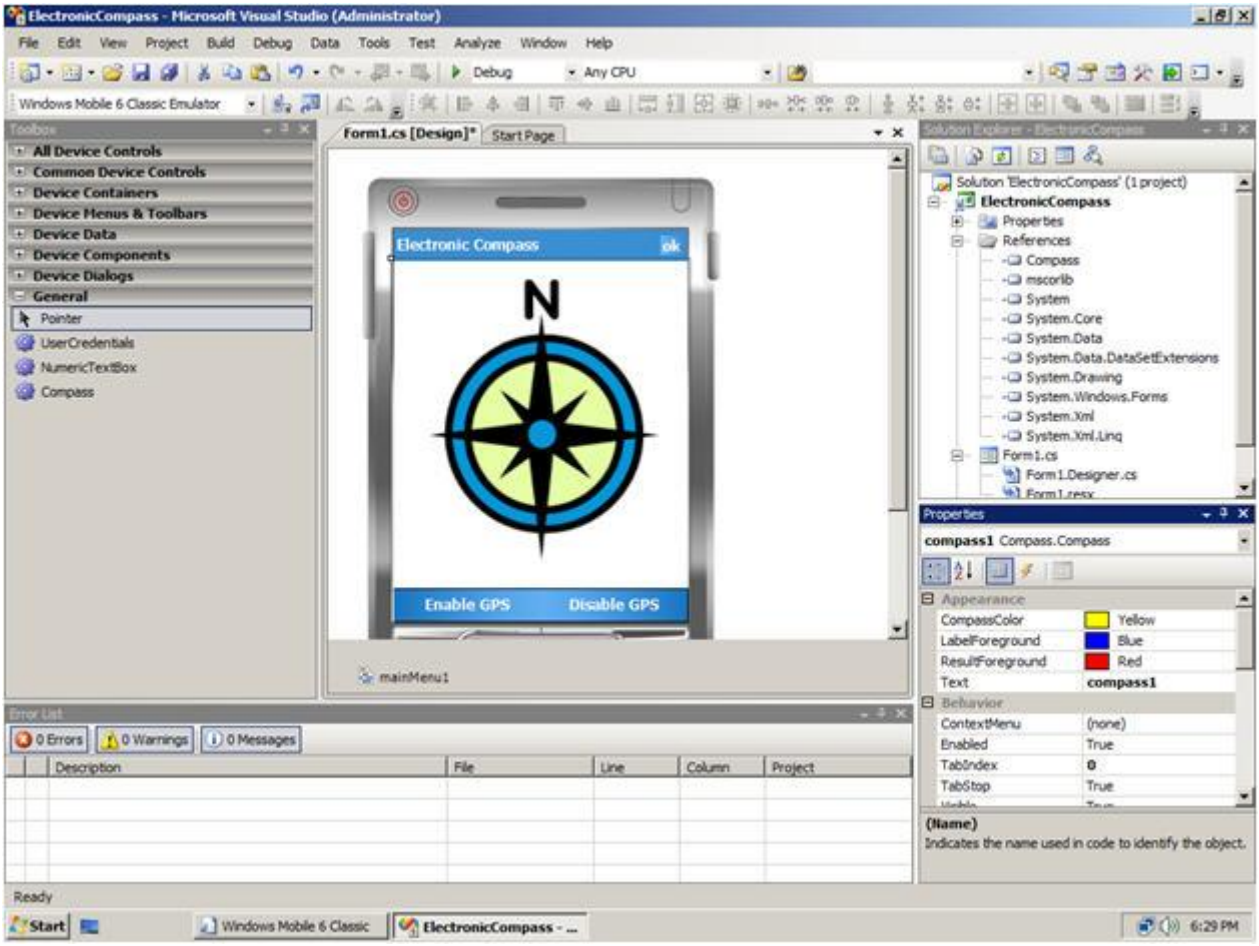
در کد قبلی فرض بر این است که شما یک نمونه از نوع Bitmap با نام memoryBitmap و همچنین یک نمونه از نوع Image با نام someImage و در نهایت یک مستطیل برای ترسیم تصویر در داخل آن ایجاد و مقدار دهی اولیه کرده اید. فراخوانی متد base.OnPaint در انتهای متد OnPaint باعث خواهد شد که آن چه در رویداد Paint از پیش از این ثبت شده است پس از نمایش محتویات توسط متد OnPaint شما، روی آن ها نمایش داده خواهد شد.

برای اضافه کردن امکانات طراحی به کنترل سفارشی تان، باید یک فایل xmta. - که نوع خاصی از فایل XML است – ایجاد نمایید. ویژوال استودیو ۲۰۰۸ با استفاده از IntelliSense (تایپ هوشمند) این امکان را به شما می دهد که یک فایل مشخصات زمان طراحی (Design-Time Attribute File) به پروژه ی تان از داخل سولوشن اکسپلورر اضافه نمایید. قطعه کد زیر یک نمونه از این نوع فایل را نشان می دهد یک مشخصه برای کنترل سفارشی شما با یک رده بندی موضوعی ایجاد می کند و در پنجره ی مشخصات ویژوال استودیو ۲۰۰۸ همراه با یک متن راهنما نشان می دهد.

```
<?xml version="1.0" encoding="utf-16"?>
<classes xmlns="http://schemas.microsoft.com/VisualStudio/2004/03/SmartDevices/XMTA.xsd">
  <class name="Compass.Compass">
    <property name="Heading">
      <category>CompassSpecific</category>
      <description>Sets the travelling speed in mph</description>
    </property>
  </class>
</classes>
```

استفاده از کنترل سفارشی داخل یک برنامه

پس از ایجاد یک کنترل سفارشی آن را داخل یک برنامه استفاده می کنید. اگر شما امکانات طراحی را برای کنترل سفارشی تان به درستی ایجاد کرده باشید و کنترل سفارشی را به جعبه ابزار ویژوال استودیو اضافه کرده باشید، قرار دادن آن در واسط کاربری برنامه تان همانند کنترل های استاندارد خواهد بود. در شکل ۴ واسط کاربر یک برنامه ی قطب نمای الکترونیکی را می بینید که از یک کنترل سفارشی Compass Control استفاده کرده است. همان طور که می بینید امکانات کامل طراحی همراه با تعدادی مشخصه برای این کنترل در دسترس قرار دارد. به علاوه می بینید که همه ی مشخصه ها مقدار پیش فرض و متن راهنما در باره ی نحوه ی استفاده از آن مشخصه دارند. برای ساختن این واسط کاربر باید یک «کنترل قطب نما» را داخل MainForm بیندازید و یک کنترل منو (Menu) هم برای فعال / غیر فعال سازی دریافت داده ها از گیرنده ی داخلی GPS به برنامه افزوده شده است.



شکل ۲۳: نرم افزار قطب نمای الکترونیکی که یک کنترل سفارشی را استفاده می کند

بعد از این که برنامه اطلاعات محل جغرافیایی GPS را دریافت کرد، جهت و سرعت حرکت کاربر در صفحه ی «کنترل قطب نما» نمایش داده می شود. برای به روز رسانی این داده ها «کنترل قطب نما» تعدادی مشخصه در اختیار برنامه نویس قرار داده است که از داخل برنامه قابل مقدار دهی هستند و مقادیری را که از گیرنده ی GPS خوانده می شوند، نشان می دهند.

افزدون اطلاعات مربوط به محل جغرافیایی به برنامه

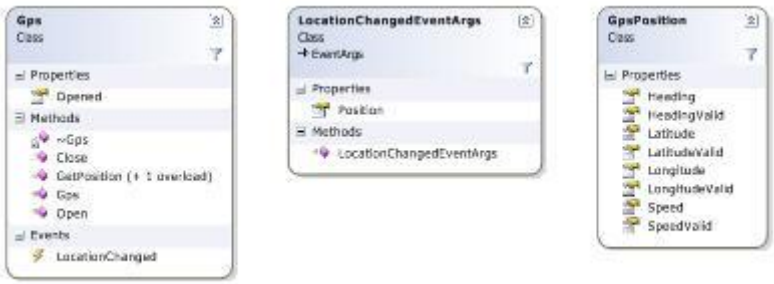
دستگاه های ویندوز موبایل ۵ و ۶ دارای یک راه انداز GPS (GPS Intermediate Driver=GPSID) هستند که برای شما بـه عنـوان یـک توسعه دهنده، دریافت اطلاعات مکان جغرافیایی از طریق GPS و همچنین به اشتراک گذاری سخت افزار آن بین برنامه های مختلف بسیار آسان خواهد بود. با گسترش دستگاه های مجهز به GPS لازم است که اطلاعاتی راجع به تولید نرم افزار بر مبنای آن داشته باشید. برای به دست آوردن اطلاعاتی در زمینه راه انداز GPS میکروسافت به آدرس زیر مراجعه نمایید:

http://msdn.microsoft.com/en-us/library/ms850332.aspx

ابزار های توسعه ی ویندوز موبایل ۵ و ۶ تعداد زیادی کد برنامه ی نمونه برای GPS دارند. یکی از این کد ها صرفا استفاده از GPSID نیست بلکـه یک کلاس مدیریتی از کارکرد های GPSID را در اختیار تان قرار خواهد داد. این مثال ها را می توانید در پوشه های زیر پیدا کنید:

<Installation Folder>\<Windows Mobile SDK>\Samples\PocketPC\Cs\Gps
<Installation Folder>\<Windows Mobile SDK>\Samples\Smartphone\Cs\Gps

پیش از استفاده از این شما باید ابتدا یک سولوشن GPS درست کنید که در یکی از پوشه های فوق، ذخیره شده است. نخسـت ویـژوال اسـتودیو ۲۰۰۸ از شما راجع به تبدیل آن به نگارش جدید خواهد پرسید چون این کدهای نمونه در ویژوال استودیو ۲۰۰۵ نوشته شده انـد و بـدون تبـدیل بـه نگـارش جدیـد کـار نخواهنـد کـرد. پـس از تبـدیل مـی توانیـد سولوشـن را درسـت کنیـد و از امکانـات فایـل اسـمبلی Microsoft.WindowsMobile.Samples.Location اسـتفاده نماییـد. بـدین منظـور بایـد ایـن اسـمبلی را در سولوشـن خودتـان وارد (import) کنید. یک مرجع به کلاس GPSID را به سولوشن تان بیفزایید تا استفاده از آن در برنامه تان امکان پـذیر گـردد. دیـاگرام داده هـا و متد های مرتبط با کلاس استفاده شده در این مقاله در شکل ۵ نمایش داده شده است.



شکل ۲۴: دیاگرام کلاس GPSID

کلاس Gps نقطه ی آغاز کار شما با سخت افزار آن است. در این کلاس متـدهایی بـرای دریافـت اطلاعـات مکـان جغرافیـایی GPS بـه صـورت بلادرنگ و هماهنگ وجود دارد و در ضمن یک رویداد که به هنگام تغییر اطلاعات مکان، فراخوانی می شود. متـدهایی نیـز بـرای فعـال/غیر فعـال سازی سخت افزار GPS هست.

پیش از دریافت اطلاعات مکانی از گیرنده ی GPS نخست لازم است که یک شیء جدید از نوع Gps ایجاد و متـد Open را بـرای آن فراخـوانی کنید. این کار باعث می شود که اگر برنامه ی شما اولین استفاده کننده ی سخت افزار GPS روی این دسـتگاه اسـت، سـخت افـزار GPS فعـال گردد. پس از پایان استفاده از آن، اگر برنامه ی شما آخرین استفاده کننده ی GPS روی دسـتگاه اسـت بایـد متـد Close را بـرای شـیء GPS فراخوانی کنیم تا سخت افزار GPS را غیرفعال کند. این کار باعث می شود که به هنگام نیـاز نداشـتن بـه GPS بـاتری دسـتگاه بیهـوده مصـرف نگردد.

قطعه کد زیر چگونگی فعال/غیر فعال سازی و اتصال به سخت افزار GPS را نشان می دهد و در مدیر رویداد های جداگانه اجرا می شود:

```
private void menuEnableGPS_Click(object sender, EventArgs e)
{
    gps.Open();
    gps.LocationChanged +=
      new LocationChangedEventHandler(gps_LocationChanged);
    menuDisableGPS.Enabled = true;
    menuEnableGPS.Enabled = false;
}
private void menuDisableGPS_Click(object sender, EventArgs e)
{
    gps.LocationChanged -= gps_LocationChanged;
    gps.Close();
    menuEnableGPS.Enabled = true;
    menuDisableGPS.Enabled = false;
}
```

اگر کاربر بخواهد برنامه را ببندد باز هم باید اتصال GPS را قطع کنید به همین دلیل باید در مدیر رویداد Closing (بستن برنامه) بررسی کنید که اگر هنوز GPS فعال است آن را غیر فعال نمایید و از فهرست رویداد LocationChanged خارج شوید (unsubscribe) و متد Close از شیء Gps را فراخوانی کنید. کار اصلی برنامه ی قطب نمای الکترونیکی در مدیر رویداد gps_LocationChanged انجام می شود. هر برنامه ای می تواند با استفاده از مدیر رویداد در آن ثبت نام (subscribe) کند. هر زمان که اطلاعات مکانی GPS تغییر کند، مدیر رویداد برنامه برای کار روی این تغییرات فراخوانی می شود. ثبت نام در رویداد LocationChanged تضمین می کند که برنامه ی شما در هر لحظه به آخرین اطلاعات مکانی دسترسی دارد، البته با این فرض که سخت افزار GPS اطلاعات ماهواره ای را به درستی دریافت می کند.

قطعه کد زیر مدیر رویداد gps_LocationChanged و نسخه ی اصلی آن را نشان می دهد:

```
void gps_LocationChanged(object sender, LocationChangedEventArgs args)
{
     GpsPosition pos = args.Position;
     compass1.HeadingValid = pos.HeadingValid;
     if (pos.HeadingValid)
     {
         compass1.Heading = pos.Heading;
     }
     compass1.SpeedValid = pos.SpeedValid;
     if (pos.SpeedValid)
     {
         compass1.Speed = pos.Speed * 1.152;    // convert knots to MPH
     }
     if (pos.HeadingValid | pos.SpeedValid)
     {
         compass1.Invalidate();
     }
}
```

با این که مشخصه های زیادی برای کلاس Gps هست، شما در این مدیر رویداد فقط مشخصه های Heading (جهت) و Speed (سرعت) را می بینید. چون نرم افزار صرفا یک قطب نمای الکترونیکی است، به همین خاطر مشخصه هایی مانند Lattitude (عرض جغرافیایی) و Longitude (طول جغرافیایی) مهم نیستند. از آن جا که اطلاعات دریافتی از ماهواره ممکن است به درستی خوانده نشوند، لازم است که درست

بودن مقادیر مشخصه های مورد نظر تان را بررسی کنید. در نهایت باید متد Invalidate فراخوانی کنید تا اگر حداقل یکی از داده های خوانده شده تغییر کرده باشد، «کنترل قطب نما» را به روز نماید.

اگر با کد بالا –که برای مدیر رویداد gps_LocationChanged نوشته اید- برنامه تان را کامپایل و اجرا نمایید با خطای زمان اجرا مواجه خواهید شد. مفهوم پیغام خطا این است که شما نمی توانید داده های «کنترل قطب نما» را به روز کنید. متن زیر را در متن پیغام خواهید دید:

Control.Invoke must be used to interact with controls created on a separate thread

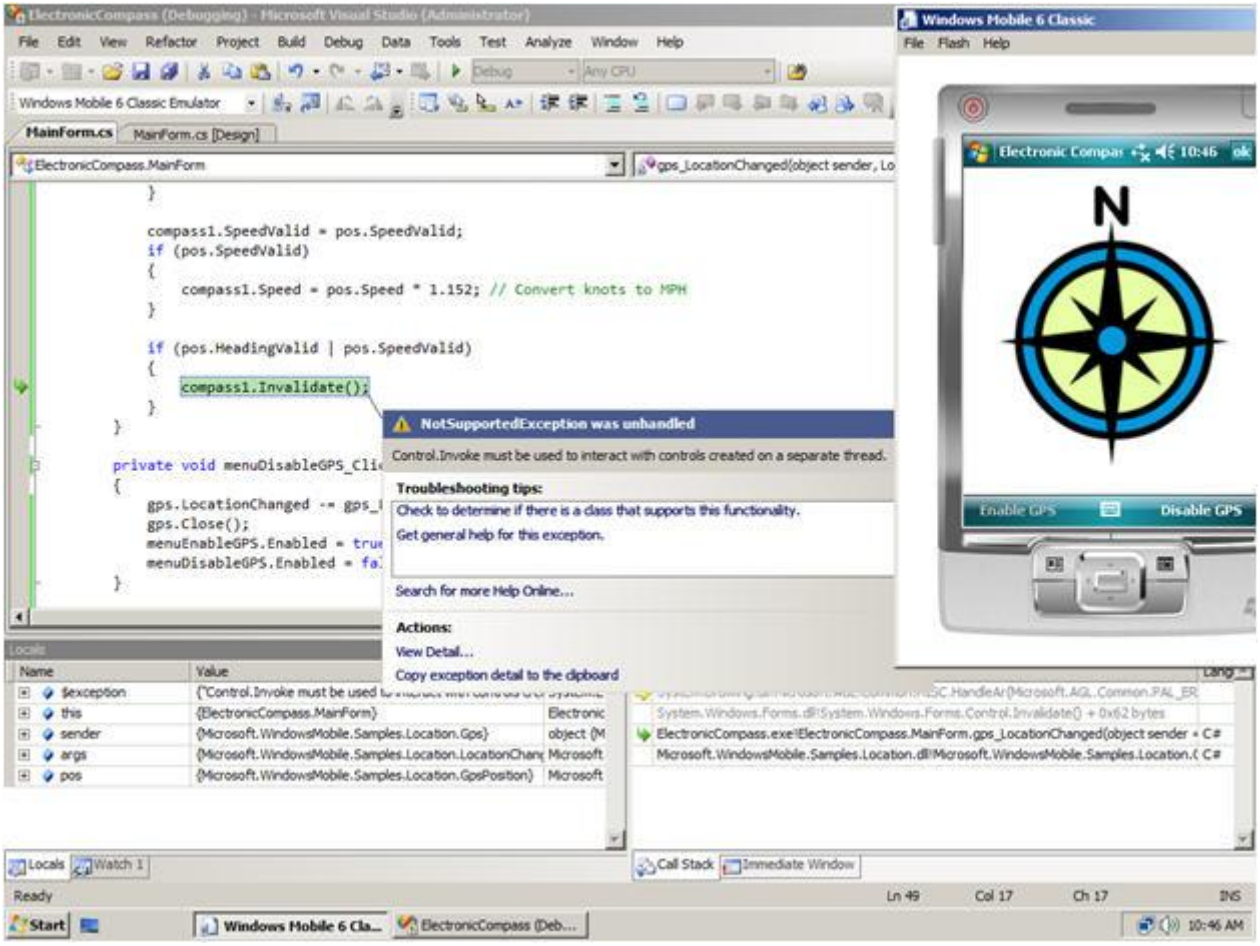
Control.Invoke باید با اشیایی در ارتباط باشد که در پردازش های مجزا ایجاد شده اند.

بازبینی پشته، اطلاعات زیر را نشان خواهد داد:

StackTrace:

```
   at Microsoft.AGL.Common.MISC.HandleAr(PAL_ERROR ar)
   at System.Windows.Forms.Control.Invalidate()
   at ElectronicCompass.MainForm.gps_LocationChanged(Object sender,
      LocationChangedEventArgs args)
   at Microsoft.WindowsMobile.Samples.Location.Gps.WaitForGpsEvents()
```

متن پیغام خطا اطلاعات مهمی برای شناسایی مشکل به ما می دهد. به نظر می رسد که چند پردازش هم زمان در برنامه ی شما اجرا می شوند، در حالی که شما فقط یک پردازش ایجاد کرده اید. اگر به Stack Trace دقت کنید متوجه می شوید که متد Invalidate –که خطا در آن رخ داده است – از داخل مدیر رویداد فراخوانی شده است که آن هم توسط یک متد داخل کلاس مدیریت GPSID فراخوانی شده است. ظاهرا GPSID یا کلاسی که آن را مدیریت می کند از چند پردازش استفاده می کنند و این همان چیزی است که باید مواظبش باشید.



شکل ۲۵: تغییر مشخصه های کنترل قطب نما (Compass) به هنگام بروز خطا

به روز رسانی کنترل های واسط کاربر در یک برنامه ی چند پردازشی

اشتباهی که بیشتر توسعه دهنده گان نرم افزار به آن دچار می شوند این است که سعی می کنند به طور مستقیم از داخل پردازش فعال، کنترل های واسط کاربر را تغییر دهند یا به آن ها دسترسی پیدا کنند. این کار منجر به واکنش غیر منتظره می شود. در نگارش ۱.۰ .NET Compact Framework برنامه پشت سر هم قفل می شود. در نگارش ۲.۰ یا بالاتر نتیجه اندکی بهتر است. هنگامی که بخواهید یک کنترل واسط کاربر را از داخل پردازش دیگری (غیر از پردازشی که کنترل مورد نظر در داخل آن ایجاد شده است) تغییر دهید خطای NotSupportedException ایجاد می گردد. در شکل ۶ شما این مورد را در برنامه ی قطب نمای الکترونیکی می بینید.

برای رفع این مشکل، باید خودتان را ملزم به رعایت این قانون بکنید که: *«فقط پردازشی (یا متدی) که کنترل واسط کاربر را ایجاد می کند می تواند آن را تغییر دهد»* زمانی که بخواهید یک کنترل را داخل پردازش فعال تغییر دهید همیشه باید متد Control.Invoke را فراخوانی کنید. این متد یک نماینده ی مخصوص را در پردازشی که به نگهدارنده ی پنجره ی (window handle) حاوی آن کنترل دسترسی دارد (و در واقع آن را ایجاد کرده است)، فراخوانی می کند. حال می توانید مدیر رویداد gps_LocationChanged را تغییر دهید. بدین منظور نخست باید یک delegate (نماینده) تعریف کنید که بتواند یک متد را به صورت آرگومان به متد Control.Invoke ارسال کند. یک delegate یک نوع داده ی ساده است که یک نشانه (اشاره گر) از متد را تعریف می کند که می تواند توسط هر متد با اشاره گر سازگار با آن مقدار دهی شود.

```
private delegate void UpdateDelegate();
void gps_LocationChanged(object sender, LocationChangedEventArgs args)
{
    GpsPosition pos = args.Position;
    compass1.HeadingValid = pos.HeadingValid;
    if (pos.HeadingValid)
    {
        compass1.Heading = pos.Heading;
    }
    compass1.SpeedValid = pos.SpeedValid;
    if (pos.SpeedValid)
    {
        compass1.Speed = pos.Speed * 1.152;
        // convert knots to MPH before displaying
    }
    if (pos.HeadingValid | pos.SpeedValid)
    {
        compass1.Invoke((UpdateDelegate)delegate()
        {
            compass1.Invalidate();
        });
    }
}
```

داخل مدیر رویداد gps_LocationChanged می بینید که compass1.Invalidate به طور مستقیم فراخوانده نمی شود بلکه از داخل متد دیگری به نام compass1.Invoke فراخوانی می شود. اگر با بحث نماینده های بی نام (anonymous delegates) آشنا نباشید این روش برای تان نا مفهوم خواهد بود. در مدیر رویداد gps_LocationChanged این قابلیت C# 2.0 برای فراخوانی compass1.Invalidate از داخل compass1.Invoke استفاده می شود. در عین حال این امکان هم هست که متد جداگانه ای برای فراخوانی compass1.Invalidate و فراخوانی آن از داخل UpdateDelegate تعریف شود. چون کنترل Compass فقط از یک جا فراخوانی می شود، استفاده از نماینده ی بی نام باعث خواهد شد که کدنویسی کوتاه تر شود.

با این تغییرات می توانید برنامه ی قطب نمای الکترونیکی را بدون خطا اجرا و اطلاعات ماهواره ای GPS را از طریق GPSID دریافت نمایید.

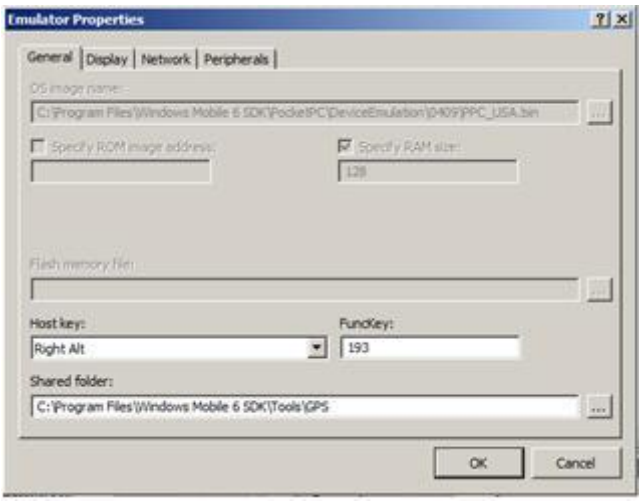
شکل ۲۶: قطب نمای الکترونیکی در حال اجرا و نمایش داده های GPS

حال این سؤال در ذهن تان هست که این نرم افزار الان داخل شبیه ساز دستگاه اجرا می شود ولی اطلاعات ماهواره ای GPS را دریافت می کند. پس چگونه کار می کند؟

آزمایش برنامه های حساس به محل جغرافیایی

اگر برنامه ای بنویسید که از GPS استفاده می کند آزمایش آن برنامه هم داستان دیگری است. اغلب گیرنده های GPS در فضای بسته یعنی همان جایی که شما الآن نشسته اید و برنامه می نویسید عمل نمی کنند. به علاوه اگر هم کار کنند اطلاعات دریافتی از ماهواره مربوط به یک محل است و شما هنگام توسعه و آزمایش برنامه نمی توانید حرکت کنید تا اطلاعات جدید را دریافت نمایید. برای حل این مشکل، ابزار توسعه ی ویندوز موبایل ۶ یک ابزار به نام FakeGPS دارد که فایل های داده ی متنی حاوی اطلاعات GPS را برای شبیه سازی کارکرد دستگاه GPS استفاده می نماید. برنامه هایی که از GPSID استفاده می کنند بدون نیاز به دستکاری می توانند از برنامه ی FakeGPS برای شبیه سازی حالت واقعی سخت افزار GPS استفاده نمایند. از آن جا که FakeGPS روی شبیه ساز دستگاه کار می کند شما می توانید نرم افزار های دیگر را نیز با استفاده از آن اجرا و آزمایش کنید.

برای استفاده از FakeGPS کافی است که آن را روی دستگاه یا شبیه ساز آن نصب نمایید. این برنامه به صورت یک فایل CAB می باشد. برای نصب آن روی شبیه ساز دستگاه کافی است که پوشه ی حاوی فایل FakeGPS را در پنجره ی Emulator Properties به اشتراک بگذارید تا همانند کارت حافظه ی جانبی عمل کند.



شکل ۲۷: اشتراک پوشه به هنگام مکان یابی شبیه سازی شده با FakeGPS

داخل شبیه ساز دستگاه می توانید با File Explorer به داخل کارت حافظه (Storage Card) بروید که به پوشه ی اشتراکی اشاره می کند. فایل CAB مربوط به FakeGPS را انتخاب و اجرا کنید تا برنامه اش در شبیه ساز نصب گردد. برنامه ی FakeGPS همراه با تعدادی فایل متنی حاوی اطلاعات GPS نصب خواهد شد. شما می توانید فایل های متنی خودتان را –که حاوی اطلاعات GPS هستند- برای حالت های مختلف تست برنامه تان به برنامه اضافه نمایید. یک فایل متنی FakeGPS می تواند چیزی شبیه به این باشد:

```
$GPGLL,4738.0173,N,12211.1874,W,191934.767,A*21
$GPGSA,A,3,08,27,10,28,13,19,,,,,,,2.6,1.4,2.3*3E
$GPGSV,3,1,9,8,71,307,43,27,78,59,41,3,21,47,0,10,26,283,40*77
$GPGSV,3,2,9,29,13,317,0,28,37,226,37,13,32,155,36,19,37,79,42*42
$GPGSV,3,3,9,134,0,0,0*46
$GPRMC,191934.767,A,4738.0173,N,12211.1874,W,0.109623,12.14,291004,,*21
$GPGGA,191935.767,4738.0172,N,12211.1874,W,1,06,1.4,32.9,M,-17.2,M,0.0,0000*75
$GPGLL,4738.0172,N,12211.1874,W,191935.767,A*21
$GPGSA,A,3,08,27,10,28,13,19,,,,,,,2.6,1.4,2.3*3E
$GPRMC,191935.767,A,4738.0172,N,12211.1874,W,0.081611,15.81,291004,,*2A
```

برای شروع دریافت داده ها از FakeGPS برنامه را در شبیه ساز اجرا کنید، سپس آن را فعال کنید و فایل داده ی GPS مورد نظر تان را انتخاب نمایید. در نهایت روی دکمه ی Done کلیک کنید تا ارسال داده ها به داخل GPSID آغاز گردد.

شکل ۹: انتخاب فایل داده ی GPS داخل برنامه ی FakeGPS

این همه ی آن چیزی بود که برای آزمایش برنامه های مبتنی بر GPS به آن نیاز داشتید. حال روی شبیه ساز دستگاه و هم روی دستگاه فیزیکی (واقعی) ویندوز موبایل می توانید برنامه تان را آزمایش کنید. پس از نصب و تنظیم FakeGPS می توانید برنامه ی قطب نمای الکترونیکی را اجرا کنید.

فصل پنجم

مقدمه ای بر SQL Server CE

مقدمه

توسعه ی نرم افزار ویندوز موبایل شباهت زیادی به توسعه ی نرم افزار در دسکتاپ دارد به ویژه زمانی که یکی از دو زبان ویژوال بیسیک یا ویـژوال سی شارپ دات نت را استفاده می کنید. شما همان ابزارهای توسعه ی برنامه های ویندوز دسکتاپ را برای ویندوز موبایل هـم اسـتفاده مـی کنیـد لیکن تفاوت هایی نیز بین این دو محیط هست. این تفاوت به هنگام استفاده از بانک اطلاعاتی هم هست. اغلب برنامه ها از بانک اطلاعاتی اسـتفاده می کنند. برای ذخیره سازی داده در دستگاه شما به عنوان توسعه دهنده دست تان باز است که از چه روشی استفاده کنید. در این مقالـه مطـالبی راجع به SQL Server 2005 Compact Edition و روش های مختلف دسترسی به داده های محلی ذخیره شده روی دستگاه خواهیـد دیـد. ما دو روش متفاوت دسترسی به داده ها را از لحاظ کارآیی زمان لازم برای اجرا و میزان کـد نوشـته شـده مقایسـه خـواهیم کـرد. در مثـال هـای کدنویسی دو جدول Orders و Order_Details از بانک اطلاعاتی نمونه ی Northwind را استفاده کرده ایم. نگارش مـورد اسـتفاده ی بانـک اطلاعاتی، SQL Server 2005 Compact Edition 3.5 همراه با ویژوال استودیو ۲۰۰۸ است.

SQL Server Compact Edition

تنوع نام ها و نگارش های SQL Server هایی که روی دستگاه های ویندوز موبایل اجرا می شوند، ممکن است شما را سر در گم کنـد. در زمـان نگارش این مقاله مهم ترین نگارش های SQL Server 2005 که در حال استفاده بر روی دستگاه های ویندوز موبایل هستند، به شرح زیر اند:

- SQL Server Mobile 3.0: همراه با ویژوال استودیو ۲۰۰۵ و SQL Server 2005 منتشر شد. این نگـارش روی کـامپیوتر هـای کوچک (Tablet PC) اجرا می شود. از زمانی که ابزارهای توسعه روی کامپیوتر های دسکتاپ نصب می شوند این نگارش روی ماشـین های خاصی اجرا می شود.
- SQL Server 2005 Compact Edition 3.1: در سال ۲۰۰۶ منتشر شد و تا حد زیـادی بـر پایـه ی SQL Server Mobile 3.0 ساخته شده است. این نگارش SQL Server CE روی دستگاه های ویندوز موبایل و در عین حال روی دسکتاپ ها و لپ تاپ هـا هم بدون محدودیت اجرا می گردد. SQL Server CE سازگاری زیادی با ویرایش های مختلف SQL Server دارد و امکانـات کامـل بانک اطلاعاتی رابطه ای را در مقیاس کوچک فراهم مـی کنـد. ایـن نگـارش SQL Server CE روی حافظـه ی اصـلی رام (ROM) دستگاه های ویندوز موبایل ۶ نصب می باشد.
- SQL Server 2005 Compact Edition 3.5 SP1: در سال ۲۰۰۷ به صورت دانلود مستقل منتشر شـد و در عـین حـال داخـل ویـــژوال اســـتودیو ۲۰۰۸ نیـــز موجـــود اســـت. بـــر مبنـــای توســـعه ی SQL Server CE 3.1 تولیـــد شـــده اســـت. این نگارش SQL Server CE امکان همسان سازی با سرور های میکروسافت را از طریق سـرویس Microsoft Syncronization برای ADO.NET فراهم می سازد.

زیر ساخت SQL Server 2005 Compact Edition

SQL Server CE یک بانک اطلاعاتی سبک است و بر خلاف نگارش های دیگر SQL Server، به صورت یک پردازش همراه با برنامه ای که از آن استفاده می کند، اجرا می شود. این بدان معنی است که SQL Server CE به صورت یک سرور مجزا پیاده سازی نشده است. بانک اطلاعاتی آن در یک فایل منفرد با پسوند sdf ذخیره می شود و همچنین امکان همسان سازی محتویات آن بین یک کامپیوتر دسکتاپ و دستگاه ویندوز موبایل از طریق برنامه ی اکتیو سینک (ActiveSync) در ویندوز ایکس پی یا برنامه ی WMDC (Windows Mobile Device Center) در ویندوز ویستا یا سون وجود دارد. امکانات پیشرفته تر همسان سازی توسط RDA در دسترس است. همچنین ادغام، کپی کردن و سرویس همسان سازی ADO.NET برای همسان سازی بین سرور و چند دستگاه و حتی پشتیبانی از سیستم تشخیص تداخل در مواردی که داده ها توسط چند کاربر به صورت هم زمان در حال به روز رسانی هستند، از جمله امکانات SQL Server CE است. این نگارش از SQL Server دارای سطوح مختلف امنیتی است و همانند بقیه، بانک های اطلاعاتی می توانند با کلمه

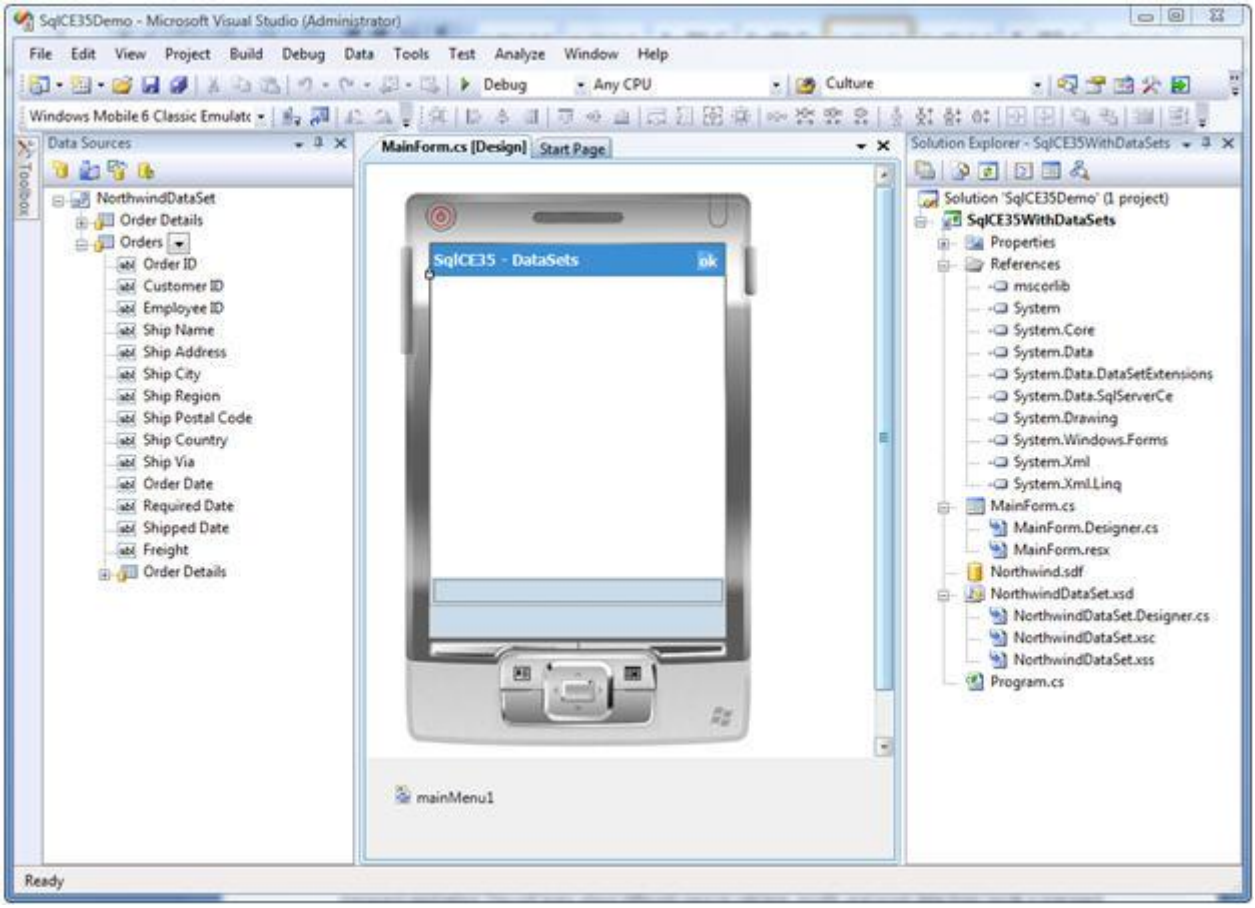
ی رمز محافظت شده، داده های داخل آنها نیز رمز نگاری شوند. این موضوع به خصوص برای دستگاه های ویندوز موبایل که در معرض گم شدن یا سرقت هستند، مهم است. به ویژه زمانی که بانک اطلاعاتی روی کارت حافظه ذخیره می شود و کارت حافظه بیشتر در معرض سرقت است.

این مقاله روی ترکیب SQL Server 2005 Compact Edition 3.5 و یک نرم افزار مدیریت شده بحث می کند. شما چند روش مختلف درج، تغییر و بازیابی داده ها را از داخل یک برنامه ی مدیریت شده –که هر کدام موافق و مخالف خود را دارد – خواهید دید.

برای مقایسه ی روش ها با یکدیگر ما از بانک اطلاعاتی Northwind جدول های Orders و Order Detail را انتخاب کرده ایم. برای استفاده از این بانک اطلاعاتی - که در مسیر C:\Program Files\Microsoft SQL Server Compact Edition\v3.5\Samples همان مسیر پیش فرض نصب ویژوال استودیو ۲۰۰۸ قرار دارد- لازم است که ویژوال استودیو ۲۰۰۸ را با مجوز مدیر سیستم (administrator elevations) در ویندوز ویستا یا سون اجرا نمایید. در غیر این صورت امکان اتصال و دسترسی به بانک اطلاعاتی را نخواهید داشت.

استفاده از دیتاست های دارای نوع

استفاده از دیتاست های دارای نوع، یک روش آسان برای دسترسی به بانک اطلاعاتی SQL Server CE است. مقدار زیادی از کد به صورت خودکار تولید می شود. لیکن دیتاست در واقع یک کپی از داده ها در حافظه ی اصلی است که از منبع داده مانند SQL Server CE استخراج شده است. به عبارت دیگر دیتاست یک کپی فیزیکی از داده هایی است که شما با آن ها کار می کنید. این بدان معنی است که زمان لازم است تا داده ها از بانک اطلاعاتی در دیتاست بارگذاری شوند و تغییرات صورت گرفته روی داده ها، به طور خودکار روی بانک اطلاعاتی اعمال نمی شوند. به هنگام مقداردهی اولیه ی دیتاست، از آن جا که شما روی کپی داده ها در حافظه ی اصلی کار می کنید، کار با داده ها بسیار سریع انجام می شود. برای کار با بانک های اطلاعاتی حجیم ممکن است با مشکل کمبود حافظه ی اصلی مواجه شوید.



شکل ۲۸: دیتاست (DataSet) های با نوع داده

در شکل ۱ ویژوال استودیو ۲۰۰۸ را می بینید که یک پروژه از نوع Smart Device همراه با یک منبع داده (Data Source) که به بانک اطلاعاتی Northwind متصل است. داخل Data Source Wizard دو جدول Orders و Order Detail انتخاب شده اند که به منبع داده ی دارای نوع متصل هستند. همان طور که در پنجره ی Data Sources می بینید، رابطه ی بین Orders و Order Detail به طور خودکار برای شما ایجاد شده است. اگر می خواهید نمایش دو جدول متصل (master-detail) را در واسط کاربر داشته باشید، باید کنترل های مورد نظر تان را از پنجره ی Data Sources روی فرم بکشید. برای نگهداری خودکار رابطه ی بین جدول ها لازم است که اطلاعات جدول Order Detail را همراه با جدول Orders انتخاب نماییم.

شکل ۲۹: فرم های تولید شده ی خودکار در محیط طراحی ویژوال استودیو

در شکل ۲ فرم هایی را که به طور خودکار ایجاد شده اند می بینید. بدون نیاز به کدنویسی می توانید اطلاعات بانک اطلاعاتی Northwind را ببینید. تنها کاری که انجام شده است، کپی داده ها داخل دیتاست در حافظه ی اصلی است. حال نیاز دارید مطمئن شوید که تغییرات در بانک اطلاعاتی به طور دائمی ذخیره شوند. کدی که توسط ویژوال استودیو ۲۰۰۸ تولید شده است فقط می تواند داده ها را در MainForm نشان دهد. (کد ۱) کد های دیگری هم جهت نمایش اطلاعات خلاصه و پنجره ی مکالمه ی ویرایش داده ها توسط ویژوال استودیو تولید شده است. پیش از مقدار دهی کنترل های واسط کاربر توسط داده های دیتاست احتمالا باید رشته ی اتصال (connection string) را برای وصل شدن به بانک اطلاعاتی SQL Server CE تغییر دهید. به ویژه برای بانک اطلاعاتی محافظت شده توسط گذرواژه، اگر افسار برنامه را به دست ویژوال استودیو بدهید، رشته ی اتصال حاوی گذرواژه خواهد بود که این خود یک ایراد امنیتی است.

```
public partial class MainForm : Form
{
   public MainForm()
   {
      InitializeComponent();
   }
   private void MainForm_Load(object sender, EventArgs e)
   {
      if (NorthwindDataSetUtil.DesignerUtil.IsRunTime())
      {
          this.order_DetailsTableAdapter.Fill(this.northwindDataSet.Order_Details);
      }
      if (NorthwindDataSetUtil.DesignerUtil.IsRunTime())
```

```
        {
            this.ordersTableAdapter.Fill(this.northwindDataSet.Orders);
        }
    }
    private void newMenuItemMenuItem_Click(object sender, EventArgs e)
    {
        ordersBindingSource.AddNew();
        SqlCE35WithDataSets.OrdersEditViewDialog ordersEditViewDialog =

            SqlCE35WithDataSets.OrdersEditViewDialog.Instance(this.ordersBindingSource);
        ordersEditViewDialog.ShowDialog();
    }
    private void ordersDataGrid_Click(object sender, EventArgs e)
    {
        SqlCE35WithDataSets.OrdersSummaryViewDialog ordersSummaryViewDialog =

            SqlCE35WithDataSets.OrdersSummaryViewDialog.Instance(this.ordersBindingSource);
        ordersSummaryViewDialog.ShowDialog();
    }
}
```

کد ۱: کد تولید شده ی خودکار برای مقدار دهی اولیه ی کنترل های واسط کاربر توسط داده های دیتاست

در مدیر رویداد MainForm_Load داده ها از بانک اطلاعاتی Northwind به دیتاست - که داده ها را توسط اتصال داده (data binding) برای کاربر نشان می دهد- کپی می شوند. کد اضافه هم برای نمایش جزئیات و افزدون سفارش (رکورد) جدید نوشته شده است. برای بررسی کارآیی این روش در بازیابی داده ها از بانک اطلاعاتی می توانید در ابتدای مدیر رویداد MainForm_Load زمان را با استفاده از شیء کرنومتر (Stopwatch) ثبت کنید و در دستور پایانی رویداد مذکور نیز کرنومتر را متوقف و زمان را دوباره ثبت کنید. حال می توانید زمان صرف شده برای بازیابی داده ها را در نوار وضعیت (status bar) نمایش دهید. به روش مشابه می توانید زمان لازم برای نمایش جزئیات را نیز به هنگام حرکت بین رکورد ها محاسبه کنید. بدین منظور زمان را به هنگام فشار دادن کلید مربوط به حرکت بین رکورد ها توسط کاربر، ثبت کرده، به محض نخستین تغییر مقدار جعبه متن در MainForm دوباره زمان را ثبت و فاصله ی زمانی را محاسبه نمایید. شکل ۳ این زمان ها را به هنگام اسفاده از دیتا ست دارای نوع داده ، به شما نشان می دهد. همان طور که می بینید زمان لازم از آغاز اجرای برنامه تا نمایش داده ها روی فرم اصلی در شبیه ساز ۱۳ ثانیه بوده است. حرکت بین رکورد ها و به ویژه ویرایش اطلاعات مربوط به جزئیات سفارش به طور میانگین ۳۰۰ میلی ثانیه زمان برده است. در عین حال شما می بینید که حرکت از رکورد اول به رکورد آخر به همان میزان حرکت از یک رکورد به رکورد بعدی اش زمان برده است و این در واقع نشان می دهد که دیتاست کاملا در حافظه ی اصلی کار می کند. اندازه گیری زمان حرکت از رکورد اول به آخر در کد ۲ نشان داده شده است.

```
private void menuGotoLast_Click(object sender, EventArgs e)
{
    ordersBindingSource.MoveFirst();
    sw.Reset();
    sw.Start();
    ordersBindingSource.MoveLast();
    sw.Stop();
    statusBar1.Text = "Goto last row: " +
                      sw.ElapsedMilliseconds + " msec.";
}
```

کد ۲: اندازه گیری زمان لازم برای حرکت از رکورد اول به آخر

شکل ۳۰: زمان لازم برای بارگزاری و نمایش داده ها و جا به جایی بین رکورد ها

از آن جا که دیتاست همه ی داده ها را در حافظه ی اصلی نگه می دارد در زمان های مشخصی لازم است که تغییرات را در بانک اطلاعاتی ثبت قطعی (commit) کنید و زمان آن بسته به نظر خود تان دارد. یک روش این است که تا جایی که ممکن است روی دیتاست کار کنید و زمانی که کاربر می خواهد برنامه را ببندد تغییرات را روی بانک اطلاعاتی ثبت قطعی نمایید. البته اگر قرار است داده ها را با یک سرور اصلی همسان سازی کنید، بهتر است که پیش از همسان سازی با آن، داده ها را ثبت قطعی نمایید. به هر حال اگر همه ی داده ها در دیتاست (حافظه ی رم) بمانند به هنگام از کار افتادن برنامه، داده های شما از بین خواهند رفت. در یک برنامه ی معمولی ممکن است شما در زمان های خاصی از جمله بسته شدن برنامه، همسان سازی با سرور خارجی و زمانی که برناه در پس زمینه قرار می گیرد، داده ها را ثبت قطعی نمایید. در عین حال ممن است بخواهید داده ها را به محض ایجاد رکورد جدید نیز ثبت قطعی کنید که در این حالت، دیتاست تقریبا به صورت یک کپی تکراری بی مصرف از داده ها خواهد بود. کد ۳ نحوه ی ثبت قطعی را به شما نشان می دهد. در این مثال در مدیر رویداد Closing (بستن برنامه) داده ها ثبت قطعی می شوند البته به شرطی که محتوای دیتاست تغییر کرده باشد.

```
private void MainForm_Closing(object sender, CancelEventArgs e)
{
   if (northwindDataSet.HasChanges())
   {
       ordersTableAdapter.Update(northwindDataSet);
       order_DetailsTableAdapter.Update(northwindDataSet);
   }
}
```

کد ۳: ثبت قطعی داده ها به هنگام بستن برنامه

زمانی که از دیتاست استفاده می نمایید، بخش اعظم کد توسط ویژوال استودیو ایجاد می شود و پس از مقدار دهی اولیه ی دیتاست کار با داده ها با سرعت انجام می شود. در واقع تنها کدی که لازم است شما بنویسید ثبت قطعی داده ها در بانک اطلاعاتی است. به علاوه ممکن است شما برای حرکت بین داده ها کدنویسی کنید و راه حل غیر از پیش فرض را برای مقدار دهی دیتاست استفاده نمایید، مثلا زمانی که بخواهید داده ها را فیلتر کرده، سپس در حافظه قرار دهید. در این مقاله ما برخی امکانات پیشرفته مانند بارگذاری داده ها با تأخیر را صرف نظر کرده ایم.

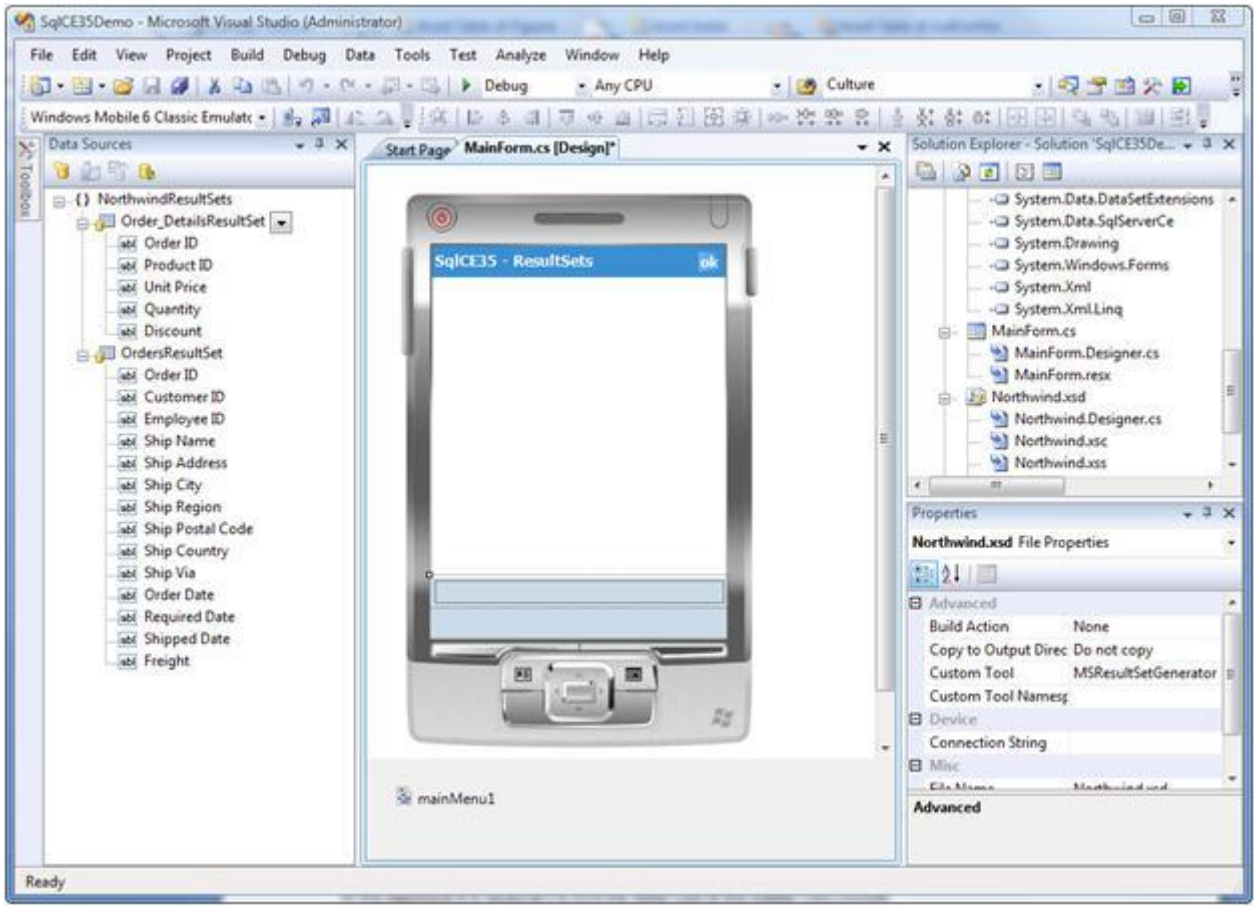
استفاده از SqlCeResultSets با نوع داده

با استفاده از SqlCeResultSets به داده های بانک اطلاعاتی SQL CE راحت تر دسترسی خواهید داشت. بخشی از کد برای شما به طور خودکار ایجاد می شود ولی نه به اندازه ای که در دیتاست دیدید. یکی از ویژگی های مهم SqlCeResultSets کار کردن مستقیم با بانک اطلاعاتی است. به عبارت دیگر داده ها در حافظه ی رم محدود و گران قیمت دستگاه ویندوز موبایل کپی نمی شوند. یک SqlCeResultSets با نوع داده، در واقع راحتی کار دیتاست (DataSet) را با کارآیی دیتاریدر (DataReader) ترکیب کرده است. برای مقایسه ی کارآیی دیتاست و SqlCeResultSets از برنامه ای مشابه - که روی همان جداول مثال قبلی با همان عملیات کار کند- استفاده خواهیم کرد. کارآیی را با همان شیء کرنومتر (Stopwatch) اندازه خواهیم گرفت. در شکل ۴ شما در پنجره ی مشخصات، یک ابزار سفارشی برای ایجاد SqlCeResultSets به جای دیتاست می بینید. مرحله ی نخست مشابه قبل است. شما یک منبع داده (DataSource) متصل به بانک اطلاعاتی Northwind را به پروژه تان می افزایید. هنگام افزودن منبع داده، یک دیتاست دارای نوع، به طور خودکار توسط ویژوال استودیو ۲۰۰۸ ایجاد می گردد. سپس شما ابزار سفارشی را در پنجره ی مشخصات تغییر می دهید تا SqlCeResultSets ها را ایجاد نماید. آن چه در شکل ۴ می بینید این است که ویژوال استودیو دو SqlCeResultSets برای تان ایجاد کرده است ولی ارتباطی بین آن دو وجود ندارد. برای داشتن یک نمایش اصلی- فرعی (Master-Detail) در MainForm باید خودتان کد مربوط به اتصال بین دو نمای اصلی و فرعی را بنویسید.

کدی که ویژوال استودیو برای نمایش داده ها در MainForm برای تان ایجاد می نماید در کد ۱ نشان داده شده است. توجه داشته باشید که فقط دیتاگرید (DataGrid) حاوی داده های جدول Orders به طور خودکار توسط ویژوال استودیو با داده های جدولش پر خواهد شد. همچنین هیچ فرم اضافه بر این ها برای نمایش خلاصه ی داده ها یا پنجره ی مکالمه ویرایش رکورد ها ایجاد نمی شود.

```
public partial class MainForm : Form
{
   private SqlCe35WithResultSets.NorthwindResultSets.OrdersResultSet
                                                   ordersResultSet;
   public MainForm()
   {
      InitializeComponent();
   }
   private void MainForm_Load(object sender, EventArgs e)
   {
      ordersResultSet = new SqlCe35WithResultSets.NorthwindResultSets.OrdersResultSet();
      ordersResultSet.Bind(this.ordersResultSetBindingSource);
   }
}
```

کد ۴: کد تولید شده ی خودکار برای مقدار دهی اولیه ی کنترل های واسط کاربر توسط SqlCeResultSet



شکل ۳۱: SqlCeResultSet های با نوع داده

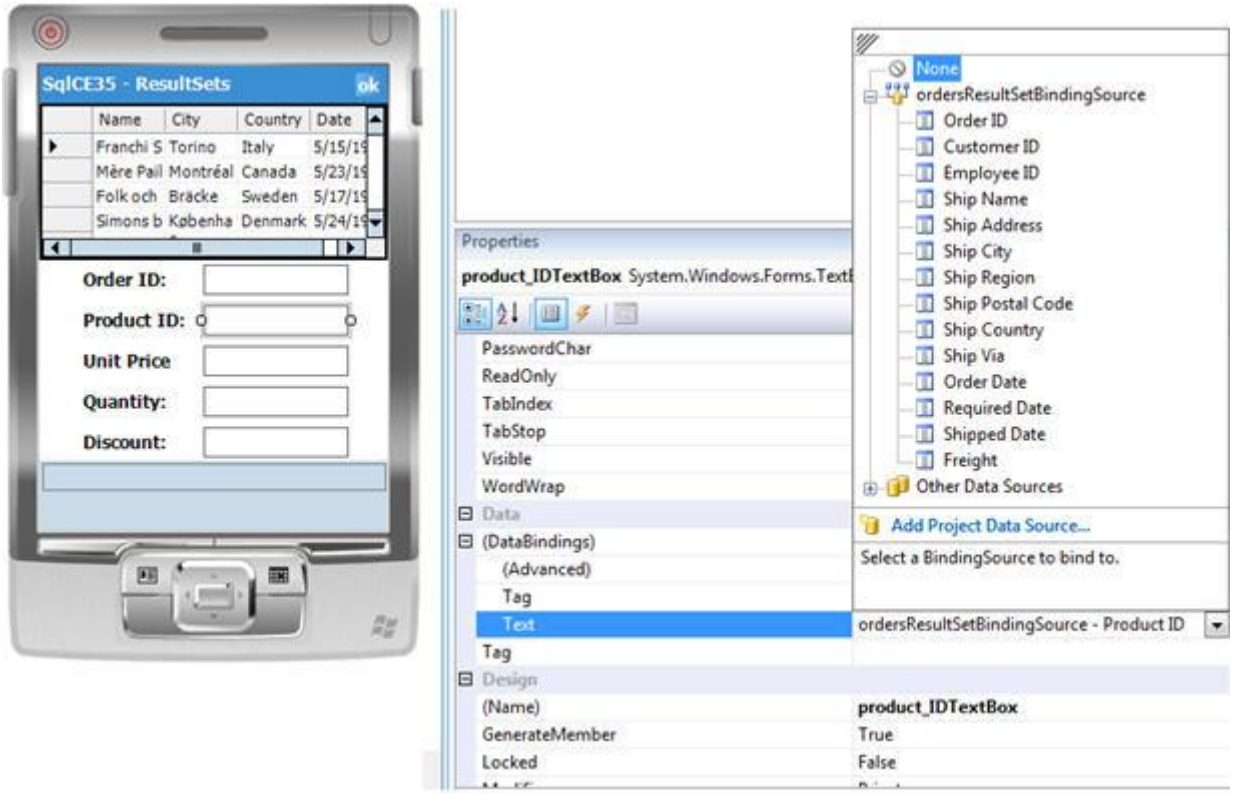
برای نمایش داده های جدول Order_Details هم زمان با رکورد انتخاب شده از جدول Order شما باید اندکی به خودتان زحمت داده، چند خط کد نویسی کنید. یک نقطه ی شروع خوب برای این کار نوشتن کد و بهبود کارکرد Order_DetailsResultSet است. روی Northwind.xsd در سولوشن اکسپلورر راست کلیک کنید. ویژوال استودیو یک فایل کد سی شارپ به نام Northwind.cs برای تان ایجاد می نماید که روی Order_DetailsResultSet کار کنید. کد تولید شده توسط محیط طراحی به صورت یک کلاس جزئی (partial) پیاده سازی می شود. در کد ۵ شما امکاناتی را که به Order_DetailsResultSet افزوده اید می بینید. یک سازنده ی (Constructor) جدید با پارامتر منطقی (boolean) ایجاد شده است که این متغیر منطقی نشان می دهد که آیا جدول Order Details نیاز به باز شدن دارد یا نه. اگر شما این سازنده را طوری فراخوانی کنید که جدول را باز نکند می توانید توسط یک روال Open به طور مجزا - که در کد ۵ برای اجرای پرس و جو روی جدول Order Details دیده می شود – جدول را باز کنید. بدین ترتیب شما یک Order_DetailsResultSet ایجاد کرده اید که فقط رکوردهای مورد نظر تان را بارگذاری کرده است. در این مقاله ما از شاخص (index) گذاری برای بالاتر بردن کارآیی و بهینه سازی بانک اطلاعاتی استفاده نکرده ایم.دلیل این کار مقایسه ی صحیح کارآیی بین دیتاست و SqlCeResultSet است. لذا با افزودن شاخص می توانید نتایج بهتری به دست آورید.

```
public partial class Order_DetailsResultSet
{
   public Order_DetailsResultSet(bool openTable)
   {
      // Create default options
      //
      resultSetOptions = System.Data.SqlServerCe.ResultSetOptions.Scrollable |
         System.Data.SqlServerCe.ResultSetOptions.Sensitive |
         System.Data.SqlServerCe.ResultSetOptions.Updatable;
      if (NorthwindUtil.DesignerUtil.IsDesignTime())
      {
         // Designtime Connection String
         resultSetConnectionString =
           "Data Source=C:\\Users\\Maarten\\Documents\\Visual Studio " +
           "2008\\Projects\\SqlCE35Demo\\SqlCE35WithDataSets\\Northwind.sdf";
      }
      else
      {
         // Runtime Connection String
         resultSetConnectionString = ("Data Source =" +
            (System.IO.Path.GetDirectoryName(
            System.Reflection.Assembly.GetExecutingAssembly().GetName().CodeBase) +
            "\\Northwind.sdf;"));
      }
      if (openTable)
          this.Open();
   }
   public void Open(string query)
   {
      System.Data.SqlServerCe.SqlCeCommand sqlCeSelectCommand = null;
      try
      {
          // Open a connection to the database
          //
          sqlCeConnection = new
              System.Data.SqlServerCe.SqlCeConnection(this.resultSetConnectionString);
          sqlCeConnection.Open();
          // Create the command
          //
          sqlCeSelectCommand = sqlCeConnection.CreateCommand();
          sqlCeSelectCommand.CommandText = query;
          sqlCeSelectCommand.CommandType = System.Data.CommandType.Text;
          // Generate the ResultSet
          //
          sqlCeSelectCommand.ExecuteResultSet(
             System.Data.SqlServerCe.ResultSetOptions.Scrollable, this);
      }
      finally
      {
```

```
            if ((sqlCeSelectCommand != null))
            {
                sqlCeSelectCommand.Dispose();
            }
        }
    }
}
```

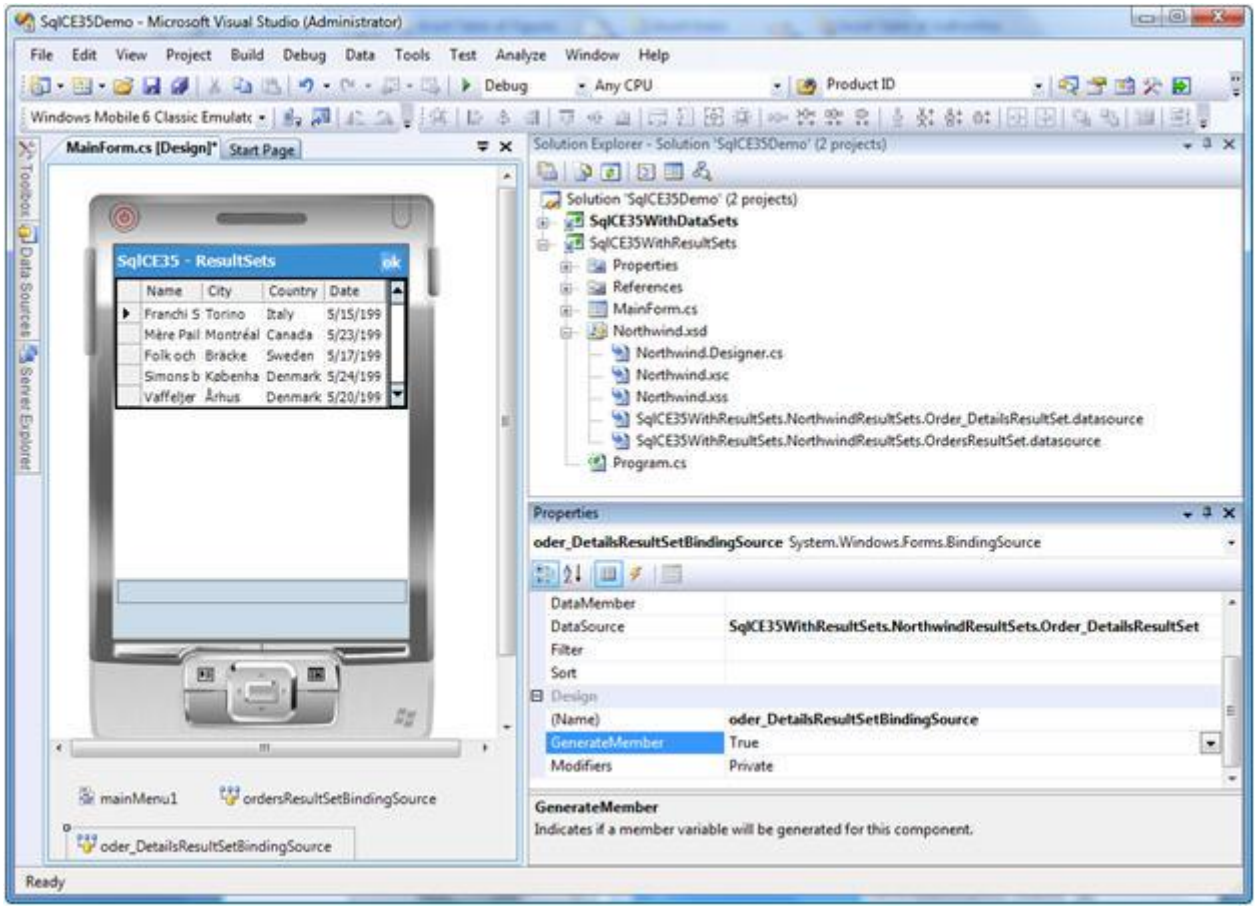
کد ۵: توسعه ی Order_DetailsResultSet

پس از توسعه ی Order_DetailsResultSet نیاز به اندکی کار بیشتر دارید. نخست این که مطمئن شوید کنترل های واسط کاربر را به یک منبع اتصال (BindingSource) جدید متصل کرده اید. زمانی که کنترل های Order_DetailsResultSet را با کشیدن روی فرم قرار می دهید، آن ها به منبع اتصالی که به هنگام کشیدن و قرار دادن دیتاگرید OrderResultSet روی فرم ایجاد شده است، متصل خواهند شد (شکل ۵). این مورد برای فیلد Order ID - که یک فیلد کلیدی مشترک بین دو جدول است- مشکلی درست نمی کند. ولی اتصال نادرست فیلد های Order_DetailsResultSet به OrderResultSetBindingSource موجب بروز خطای زمان اجرا خواهند شد، چون در این منبع داده تعریف نشده اند.



شکل ۳۲: اتصال جعبه متن Product ID به منبع داده ی نادرست

راحت ترین راه رفع این مشکل افزودن یک منبع اتصال (binding source) به سولوشن است که مشخصه ی DataSource آن را برابر با Order_DetailsResultSet قرار دهیم (شکل ۶). پس از این کار به هنگام کشیدن و انداختن کنترل های واسط کاربر از Order_DetailsResultSet به روی MainForm ویژوال استودیو از شما خواهد پرسید که کدام منبع اتصال را می خواهید استفاده نمایید. با مشخص کردن منبع اتصالی که خود تان اضافه نموده اید، این ایراد برطرف خواهد شد. مورد بعدی، نوشتن کد برای نمایش جزئیات اطلاعات رکورد انتخاب شده ی فعلی از دیتاگرید است. با چند خط کد نویسی در MainForm می توانید این کار را تمام کنید. از همان بخش توسعه ی Order_DetailsResultSet نوشته شده در کد ۵ استفاده کنید.



شکل ۳۳: افزودن دستی منبع اتصال (binding source) و انتساب آن به Order_DetailsResultSet

هر بار که کاربر یک رکورد (سفارش) را از دیتاگرید انتخاب می کند (شکل ۶)، ابتدا باید مقدار فیلد (کلید اولیه) Order ID را به دست آورید. سپس با استفاده از آن یک عبارت پرس و جو درست کنید که از جدول Order Details اطلاعات مربوط به جزئیات سفارش انتخاب شده را به دست آورد. پس از این که یک Order_DetailsResultSet ایجاد و رکورد منطبق با شرایط پرس و جو، در آن ذخیره شد، باید منبع اتصال (BindingSource) را به آن متصل کنیم و سپس ResultSet اصلی را از حافظه خارج کنیم. در کد ۶، روش اتصال دو جدول Orders و OrderDetails را از طریق کد نویسی می بینید. این کد در هر بار تغییر اشاره گر رکورد (فعلی) داخل دیتاگرید فراخوانی می شود.

```
private void ordersResultSetBindingSource_PositionChanged(object sender, EventArgs e)
{
   SqlCE35WithResultSets.NorthwindResultSets.Order_DetailsResultSet orgDetailsRS=

       order_DetailsResultSet;
   GetOrderDetails();
   if (orgDetailsRS != null)
      orgDetailsRS.Dispose();
}
private void GetOrderDetails()
{
   int orderID =
    (int)((RowView)this.ordersResultSetBindingSource.Current).UpdatableRecord["Order ID"];
   string query = "SELECT * FROM [Order Details] WHERE [Order ID] = '" + orderID +"'";
```

```
    order_DetailsResultSet =
      new SqlCE35WithResultSets.NorthwindResultSets.Order_DetailsResultSet(false);
    order_DetailsResultSet.Open(query);
    order_DetailsResultSet.Bind(order_DetailsResultSetBindingSource);
}
```

کد ۶: اتصال نمای فرعی (details) به اصلی (master)

زمانی که کاربر یک رکورد جدید را (در دیتاگرید جدول Orders) انتخاب می کند رویداد PositionChanged روی منبع اتصال ordersResultSetBindingSource فرا خوانده می شود. این رویداد می تواند برای به دست آوردن مقدار فیلد Order ID رکورد فعلی استفاده شود. حافظه ی order_DetailsResultSet فعلی آزاد شده، یکی دیگر از آن با استفاده از یک پرس و جوی جدید بر مبنای Order ID ساخته می شود که اطلاعات مربوط به جزئیات سفارشی را که شماره ی آن برابر با Order ID فعلی است، از جدول Order Details بر می گرداند.

در بخش دیتاست دیدید که فرم نمایش خلاصه ی داده ها و ویرایش رکورد به طور خودکار برای تان ایجاد می شود. لیکن در مورد SqlCeResultSet ویژوال استودیو چنین کاری نمی کند و شما خودتان باید فرم ها و کد های مربوط به آن را ایجاد کنید. در این مقاله به دلیل این که ما در باره ی مقایسه ی کارآیی دو روش صحبت می کردیم این بخش را انجام ندادیم. مورد دیگر این که چون SqlCeResultSet به طور مستقیم داده ها را روی بانک اطلاعاتی ذخیره می کند، به هنگام تغییر داده ها نیازی به کدنویسی برای ثبت قطعی داده ها (همانند دیتاست) نیست.

مشابه با روش دیتاست در این جا هم برای اندازه گیری کارآیی زمانی، از شیء کرنومتر استفاده می کنیم. در مدیر رویداد MainForm_Load می توانیم زمان بارگذاری داده ها را اندازه گرفته در نوار وضعیت نمایش دهیم. به همین ترتیب زمان لازم برای به دست آوردن اطلاعات مربوط به جزئیات سفارش را هنگام حرکت از یک رکورد به رکورد دیگر به دست می آوریم. شکل ۷ نتایج این اندازه گیری کارآیی را برای SqlCeResultSet نشان می دهد. همان طور که می بینید از لحظه ی بارگذاری برنامه در شبیه ساز دستگاه تا بازیابی داده ها از بانک اطلاعاتی ۳۰۰ میلی ثانیه زمان لازم است. حرکت بین رکورد ها، به روز رسانی آنها به ویژه جزئیات سفارش (Order details) ۵۰ میلی ثانیه زمان لازم دارد. حرکت از اولین رکورد به آخرین رکورد اندکی بیشتر از حالت دیتاست زمان لازم دارد چون تعداد رکوردهای بیشتری باید خوانده شوند.

شکل ۳۴: زمان لازم برای بارگزاری و نمایش داده ها و جا به جایی بین رکورد ها (مقایسه کنید با شکل ۳)

نتیجه گیری

استفاده از دیتاست بسیار آسان است. از آن جا که ویژوال استودیو ۲۰۰۸ خودش کد را تولید می کند شما کد نویسی چندانی لازم ندارید. به ویژه در حالت نمایش هم زمان جدول های اصلی- فرعی مناسب است. به دلیل بارگذاری همه ی داده ها به حافظه اصلی، زمان بیشتری برای نمایش اولیه ی رکورد ها در فرم برنامه ی شما لازم دارد و همچنین حافظه ی اصلی نازنین و گران قیمت دستگاه ویندوز موبایل تان را هم باید مصرف کنید.

SqlCeResultSet کارآیی بیشتری دارد به ویژه هنگام بارگذاری اولیه ی داده ها. چون به طور مستقیم روی بانک اطلاعاتی کار می کند و فقط رکورد هایی را که در دیتاگرید نشان می دهد، به حافظه ی اصلی بارگذاری می کند. در حالت نمایش هم زمان دو جدول اصلی- فرعی نیاز به کد نویسی دارید. سرعت SqlCeResultSet با ایجاد شاخص (index) در بانک اطلاعاتی می تواند بیشتر شود. اگر فرم نمایش ویرایش و خلاصه ی داده ها را می خواهید باید خودتان کد نویسی کنید. در نهایت کارآیی بالاتر این روش به زحمتش می ارزد که چند خط کد را بنویسید و شاید هم از یک مقاله کپی کنید. **(;**

فصل ششم

امنیت دستگاه و نصب نرم افزار

مقدمه

توسعه ی نرم افزار ویندوز موبایل شباهت زیادی به توسعه ی نرم افزار در دسکتاپ دارد به ویژه زمانی که یکی از دو زبان ویژوال بیسیک یا ویژوال سی شارپ دات نت را استفاده می کنید. شما همان ابزارهای توسعه ی برنامه های ویندوز دسکتاپ را برای ویندوز موبایل هم استفاده می کنید لیکن تفاوت هایی نیز بین این دو محیط هست. بسته به شرکت سازنده یا اپراتور تلفن همراه که دستگاه ویندوز موبایل را عرضه می کنند، این دستگاه ها دارای تنظیمات امنیتی متنوعی هستند. از دیدگاه یک توسعه دهنده امنیت دستگاه تعیین کننده ی اجرا شدن یا نشدن یک برنامه روی آن و همچنین منابعی از سیستم است که برنامه به آن ها دسترسی دارد. این مسأله در چگونگی توزیع یا نصب نرم افزار به ویژه زمانی که با دستگاه (گوشی) های متنوعی از سازنده گان مختلف سر و کار دارید، اهمیت بیشتری پیدا می کند.
این مقاله اطلاعات مفیدی را در زمینه ی امنیت دستگاه های ویندوز موبایل، چگونگی آزمایش تنظیمات مختلف امنیتی با استفاده از ابزار مدیر امنیت (security manager) ویژوال استودیو ۲۰۰۸ و در نهایت نحوه ی نصب درست برنامه ها روی دستگاه های ویندوز موبایل به شما ارایه می دهد.

امنیت در دستگاه های ویندوز موبایل

دستگاه های مختلف ویندوز موبایل بسته به شرکت سازنده شان تنظیمات امنیتی متنوعی دارند. یک دستگاه ممکن است به طور کامل نصب هر برنامه ای را بپذیرد و دستگاه دیگر ممکن است به هنگام اولین نصب یا اجرای یک برنامه از کاربر تأییدیه بگیرد. یا این که دستگاه به کلی در برابر نصب برنامه هایی که تأییدیه ی امنیتی از سازنده ی آن ندارد، مسدود باشد.

یک دلیل وجود تنظیمات امنیتی مختلف این است که اغلب دستگاه های پیشرفته ی ویندوز موبایل دارای امکانات تلفن همراه هستند و برای ارتباط در یک شبکه ای که در اختیار اپراتور تلفن همراه است، به کار می روند. لذا یکی از دلایل دستگاه ویندوز موبایل امنیت خود شبکه است. در واقع اپراتور های تلفن همراه می خواهند شبکه شان را در برابر نرم افزار هایی که به راحتی توسط کاربران نصب می شوند، محافظت کنند. به همین دلیل نمی توان هر برنامه ای را - حتی اگر مخصوص ویندوز موبایل نوشته شده باشد- روی یک دستگاه ویندوز موبایل نصب کرد.

بدین ترتیب برای تأمین امنیت مورد درخواست اپراتور های تلفن همراه، دستگاه های ویندوز موبایل دارای تنظیمات امنیتی مختلفی هستند. لیکن امنیت اعمال شده در یک دستگاه ویندوز موبایل صرفا به دلیل اپراتور تلفن همراه یا شرکت استفاده کننده ی آن نیست، بلکه به نوع استفاده ی آن نیز بستگی دارد. امنیت دستگاه های ویندوز موبایل ضروری و به طور کامل داخل سیستم عامل تعبیه شده است. لایه های مختلف امنیتی با هم ترکیب شده اند تا در نهایت بر مبنای یک تنظیم امنیتی مشخص تعیین کنند که آیا برنامه ها می توانند روی دستگاه نصب و اجر شوند یا نه و اگر پاسخ به این پرسش مثبت است، چگونه؟

مجوزهای نصب نرم افزار

بسته به تنظیمات امنیتی یک دستگاه خاص ویندوز موبایل برنامه ها ممکن است روی آن اجرا شوند یا این که به کلی اجرای آن ها ممنوع باشد. مجوز های زیر برای اجرای برنامه ها روی یک دستگاه ویندوز موبایل تعریف شده اند:

ویژه (Privileged): نرم افزار هر کاری روی دستگاه می تواند انجام دهد، دسترسی کامل به سیستم فایل و رجیستری سیستم دارد و همچنین می تواند تأییدیه برای اجرای نرم افزار های دیگر روی دستگاه نصب کند.

عادی (Normal): نرم افزار محدودیت هایی برای اجرا دارد. توابع Win32 API را نمی تواند فراخوانی کند، نمی تواند در بخش های محافظت شده ی رجیستری تغییر دهد یا روی فایل های سیستم بنویسد یا تأییدیه نصب کند.

مسدود (Blocked): نرم افزار به هیچ وجه مجوز اجرا ندارد.

تأییدیه های نرم افزار

برای به دست آوردن مجوز اجرا برای نرم افزار تان می توانید برای آن نشان تأییدیه بگیرید. دستگاه های ویندوز موبایل دو نوع تأییدیه دارند. تأییدیه ی ویژه و تأییدیه ی عادی که مطابق با نوع تأییدیه سطح دسترسی آن هم تعیین می شود. از آن جا که یک گروه خاص (مانند شرکت سازنده یا اپراتور تلفن همراه) این تأییدیه ها را ارایه می دهند، شما باید با این مراکز برای دریافت تأییدیه ی نرم افزار تان در ارتباط باشید. اگر چه همانند یک ISV (فروشنده ی مستقل نرم افزار = Independent Software Vendor) ممکن است بخواهید که نرم افزار تان روی دستگاه های متنوعی کار کند. شرکت Mobile2Market چنین کاری را آغاز کرده است و هر تولید کننده ی نرم افزار می تواند محصولش را برای دریافت تأییدیه به آن ها ارایه نماید. دریافت تأییدیه از Mobile2Market به این معنا است که شما یا سازمان شما به عنوان یک تولید و توزیع کننده ی نرم افزار به رسمیت شناخته می شوید.

سطوح دسترسی نرم افزار

سطوح دسترسی مختلف مشخص می کنند که یک نرم افزار بدون تأییدیه چه کارهایی می تواند روی دستگاه ویندوز موبایل انجام دهد. این سطوح دسترسی مختلف، لایه (tier) نامیده می شوند. ویندوز موبایل دو سطح دسترسی به شرح زیر دارد:

امنیت یک لایه (One-tier security): نرم افزار دارای تأییدیه با مجوز ویژه روی دستگاه اجرا می شود و دسترسی کامل به همه ی بخش های دستگاه دارد. نرم افزار بدون تأییدیه ممکن است بسته به تنظیمات امنیتی دستگاه بتواند روی آن اجرا شود. در این صورت با همان مجوز ویژه اجرا خواهد شد.

امنیت دو لایه (Two-tier security): نرم افزار دارای تأییدیه می تواند روی دستگاه اجرا شود و بسته به نوع تأییدیه اش می تواند با مجوز عادی یا ویژه اجرا شود. نرم افزار بدون تأییدیه بسته به تنظیمات امنیتی دستگاه ممکن است اجرا شود، لیکن در صورت اجرا با مجوز عادی اجرا خواهد شد.

تنظیمات امنیتی دستگاه

حال که تفاوت مجوز اجرای نرم افزار با نشان تأییدیه ی نرم افزار را متوجه شدید، یک پرسش مهم می ماند. آیا نرم افزار شما با وجود نداشتن تأییدیه می تواند روی یک دستگاه خاص ویندوز موبایل نصب و اجرا شود؟ چند تنظیم امنیتی استاندار برای دستگاه های ویندوز موبایل تعریف شده است که برخی به شما این مجوز را خواهند داد که برنامه ی بدون تأییدیه را اجرا نمایید:

- بدون امنیت (Security Off): همه ی نرم افزار ها صرف نظر از داشتن یا نداشتن تأییدیه می توانند با مجوز ویژه روی دستگاه اجرا شوند.
- تأیید یک لایه ای توسط کاربر (One-Tier Prompt): همه ی نرم افزار های دارای نشان تأییدیه ی امنیتی، با مجوز ویژه روی دستگاه اجرا می شوند. اجرای نرم افزار های بدون تأییدیه با تصمیم کاربر صورت می گیرد. اگر کاربر اجرای نرم افزار بدون تأییدیه را بپذیرد، نرم افزار با مجوز ویژه اجرا خواهد شد.
- تأیید دو لایه ای توسط کاربر (Two-Tier Prompt): همه ی نرم افزار های دارای تأییدیه روی دستگاه اجرا خواهند شد لیکن مجوز اجرای ویژه یا عادی بسته به نوع تأییدیه ی امنیتی تعیین می شود. نرم افزار های بدون تأییدیه، با اجازه ی کاربر می توانند اجرا شوند لیکن در صورت اجرا، فقط با مجوز عادی اجرا خواهند شد.
- تأییدیه شخص ثالث (Third-Party Signed): فقط نرم افزار های دارای تأییدیه ی امنیتی می توانند روی دستگاه اجرا شوند. مجوز اجرای ویژه یا عادی بسته به نوع تأییدیه ی نرم افزار خواهد بود. برای دریافت تأییدیه باید در یک سیستم توسعه ی نرم افزار مانند Mobile2Market ثبت نام کنید. اغلب دستگاه های ویندوز موبایل که توسط اپراتور های تلفن همراه خریداری و ارایه می شوند تأییدیه ی Mobile2Market را با خود دارند.
- مسدود (Locked): فقط نرم افزار های دارای تأییدیه ی امنیتی می توانند روی دستگاه اجرا شوند. مجوز اجرای ویژه یا عادی بر حسب نوع تأییدیه مشخص می شود.

مجوز اجرا و فایل های کتابخانه ی DLL

یک مورد دیگر هم هست که باید فکری به حال آن بکنیم و آن هم فایل های کتابخانه ی دینامیکی است که از این پس از همان اصطلاح آشنای DLL را استفاده می کنیم. همانند برنامه ها DLL ها هم می توانند بدون تأییدیه باشند یا این که تأییدیه با مجوز اجرای ویژه یا عادی داشته باشند. از آن جا که DLL ها از داخل برنامه ها استفاده می شوند یک سؤال جالب پیش می آید. اگر مثلا یک برنامه با مجوز دسترسی ویژه یک DLL بدون تأییدیه را که آن هم از داخل خود یک تابع مورد اطمینان Win32 API را فرا می خواند، استفاده کند چه اتفاقی می افتد؟ اگر فایل DLL هم به همان سطح دسترسی برنامه ارتقا داده شود که این یک نقص امنیتی خواهد بود.

از این رو قوانین زیر برای برنامه ها ایجاد شده است تا مطابق آن به DLL ها دسترسی پیدا کنند:

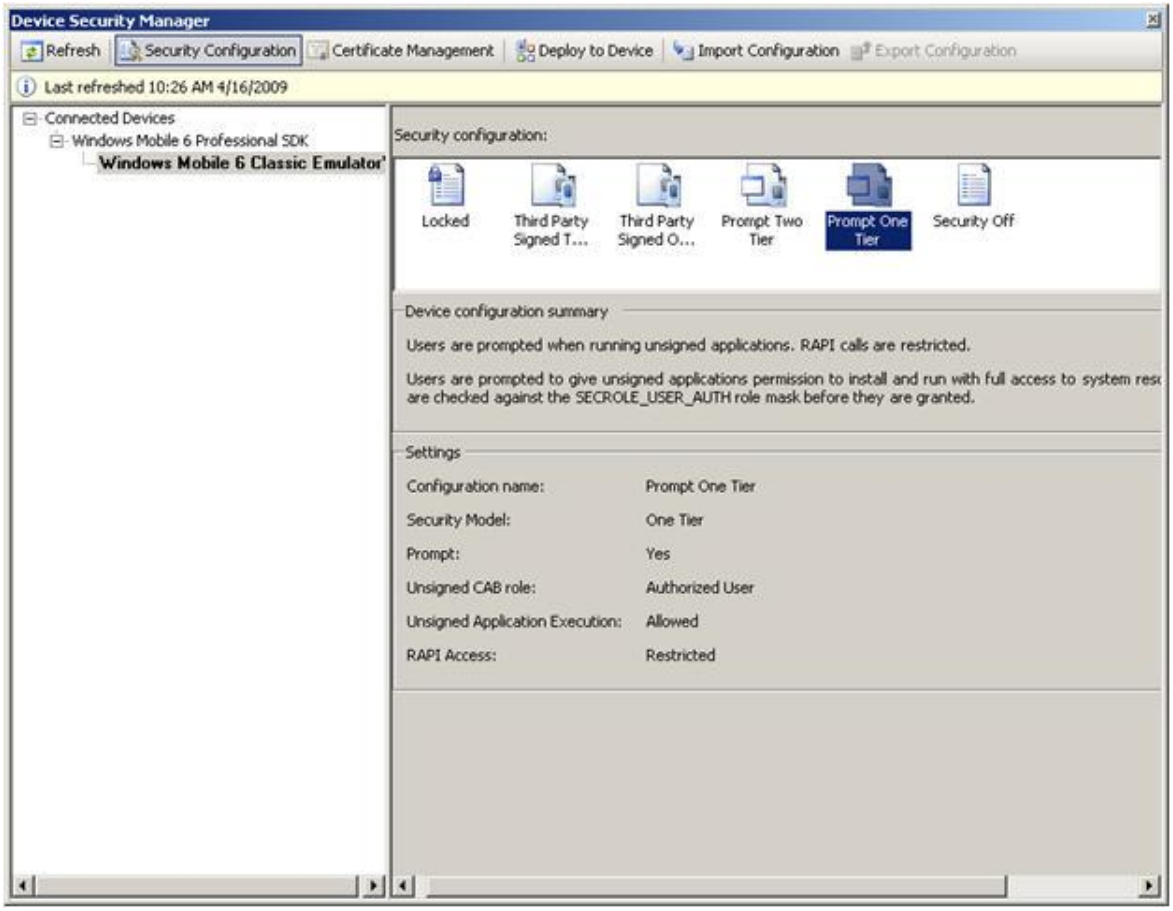
ترکیب برنامه و DLL	بدون امنیت	تأییدیه ی یک لایه ای	تأییدیه ی دو لایه ای
برنامه ی بدون مجوز			
DLL بدون مجوز	ویژه	ویژه با اجازه ی کاربر	عادی با اجازه ی کاربر
DLL با دسترسی عادی	ویژه	ویژه با اجازه ی کاربر	عادی با اجازه ی کاربر
DLL با دسترسی ویژه	ویژه	ویژه با اجازه ی کاربر	عادی با اجازه ی کاربر
برنامه ی با مجوز عادی			
DLL بدون مجوز	ویژه	ویژه	عادی با اجازه ی کاربر
DLL با دسترسی عادی	ویژه	ویژه	عادی
DLL با دسترسی ویژه	ویژه	ویژه	عادی
برنامه ی با مجوز ویژه			
DLL بدون مجوز	ویژه	ویژه	غیر مجاز
DLL با دسترسی عادی	ویژه	ویژه	غیر مجاز
DLL با دسترسی ویژه	ویژه	ویژه	ویژه

آزمایش برنامه در حالت های مختلف تنظیمات امنیتی

یکی از ابزار های ویژوال استودیو ۲۰۰۸ «مدیر امنیت دستگاه» یا همان Device Security Manager است که به شما امکان می دهد که تنظیمات امنیتی فعال روی یک دستگاه ویندوز موبایل شبیه سازی شده را به دست آورید.

نکته: شما زمانی می توانید تنظیمات امنیتی را روی یک دستگاه فیزیکی ویندوز موبایل تغییر دهید که تنظیمات امنیتی فعلی آن، به شما اجازه ی این کار را بدهد. دقت کنید که اگر تنظیمات امنیتی یک دستگاه را به حالت مسدود (Locked) تغییر دهید، این حالت قابل بازگشت نیست. پس در این مورد احتیاط کنید. البته مدیر امنیت دستگاه در این حالت به شما هشدار خواهد داد. با استفاده از شبیه ساز دستگاه به راحتی می توانید تمامی حالت های امنیتی را آزمایش نمایید.

با استفاده از برنامه ی Device Security Manager می توانید هر تنظیم امنیتی را که از پیش تعریف شده است، فعال کنید یا تنظیم امنیتی جدید را وارد (import) برنامه کنید. به علاوه می توانید فهرست تمامی تأییدیه های امنیتی نصب شده روی دستگاه ویندوز موبایل یا شبیه ساز دستگاه را به دست آورید.



شکل ۳۵: برنامه ی Device Security Manager

تنظیمات امنیتی دستگاه ها می تواند داخل پرونده های XML تعریف شوند. مثال زیر را داخل ویژوال استودیو ۲۰۰۸ می توانید ببینید:

```
<wap-provisioningdoc />
  <characteristic type="SecurityPolicy" />
    <parm name="4102" value="1" />
    <parm name="4122" value="0" />
    <parm name="4123" value="0" />
  </characteristic />
</wap-provisioningdoc />
```

آزمایش سطوح مختلف دسترسی در تنظیمات امنیتی مختلف

حال که یاد گرفتید چطور تنظیمات امنیتی مختلف را اعمال کنید، چند آزمایش را انجام می دهیم. قطعه کد زیر به عنوان یک مثال ساده می تواند نقطه ی آغاز باشد. این قطعه کد از یک تابع تأیید شده ی Win32 API استفاده می کند. اگر شما این قطعه کد را در یک برنامه یا DLL قرار دهید و برای هر کدام تأییدیه (certificate) های مختلف بگیرید، یا حتی بدون تأییدیه اجرا کنید در نهایت خواهید توانست کارکرد برنامه را در تنظیمات مختلف امنیتی آزمایش کنید.

```
[DllImport("coredll.dll")]
public extern static void PowerOffSystem();
private void btnInstructions_Click(object sender, EventArgs e)
{
    PowerOffSystem();
}
```

اگر برنامه مجاز به فراخوانی API های تأیید شده باشد، با اجرای آن می بینید که بلافاصله شبیه ساز دستگاه بسته می شود چون خاموش شده است و اگر برنامه مجاز به این فراخوانی نباشد هیچ اتفاقی نمی افتد و API نادیده گرفته می شود. اگر شما بخواهید در نواحی محافظت شده ی سیستم فایلی یا رجیستری دستگاه بنویسید با خطای دسترسی غیر مجاز (UnauthorizedAccessException) مواجه خواهید شد.

نصب برنامه

تظیمات امنیتی دستگاه روی نصب برنامه ها هم اثر می گذارد. تا اینجای مقاله، فرض بر این بوده است که نصب نرم افزار ها از داخل ویژوال استودیو ۲۰۰۸ نصب می شوند و تنظیمات امنیتی نیز از طریق Device Security Manager صورت می گیرد. صد البته هنگامی که در حال برنامه نویسی هستید این امکان، کار شما را خیلی آسان می کند، لیکن در نهایت شما برنامه نان را منتشر خواهید کرد و به همین دلیل باید تجربه مطلوبی در زمینه ی نصب برنامه برای کاربران نهایی (end users) داشته باشید و این یعنی برنامه ی شما به صورت یک فایل CAB یا MSI منتشر گردد. اگر برنامه تان به صورت فایل CAB منتشر شود، کاربر باید فایل را به داخل دستگاه ویندوز موبایل خودش کپی کند و توسط برنامه ی اکسپلورر آن را باز و اجرا نماید. اگر هم بخواهید فایل CAB را داخل یک فایل Microsoft Installer (یا همان MSI) قرار دهید کاربران می توانند از طریق ویندوز کامپیوتر دسکتاپ خودشان زمانی که دستگاه ویندوز موبایل شان از طریق برنامه ی ActiveSync در ویندوز ایکس پی یا برنامه ی Windows Mobile Device Center در ویندوز ویستا یا سون به کامپیوتر متصل است، برنامه را نصب کنند.
اگر تنظیمات امنیتی دستگاه سختگیرانه باشد ممکن است اجازه ندهد که فایل CAB بدون امضا روی آن نصب شود. در واقع صرف تأییدیه داشتن برنامه تان کافی نیست بلکه باید برای فایل CAB هم امضای تأییدیه بگیرید. ابزار توسعه ی ویندوز موبایل ۶ یک برنامه ی خط فرمانی به نام cabsigntool.exe برای این کار دارد که از signtool.exe ی ویژوال استودیو ۲۰۰۸ استفاده می کند. برای راحتی کار بهتر است از خط فرمان ویژوال استودیو ۲۰۰۸ استفاده کنید که مجبور نباشید متغیر محیطی path را برای یافتن این برنامه مقدار دهی کنید. مثال زیر نحوه ی استفاده ی cabsigntool.exe را از طریق خط فرمان به شما نشان می دهد:

```
cabsigntool "C:\Users\UserVS2008\Documents\Visual Studio
2008\Projects\HOL6SampleApp\DVDsMobileCAB\Debug\DVDsMobileCAB.cab"
"C:\Users\UserVS2008\Documents\Visual Studio
2008\Projects\HOL6SampleApp\DVDsMobileCAB\Debug\DVDsMobileCAB.cab"
-f "C:\Program Files\Windows Mobile 6 SDK\Tools\Security\SDK Development
Certificates\SampleUnPrivDeveloper.pfx"
```

در این مثال فایل CAB با نام DVDsMobileCab.cab و تمام برنامه های اجرایی داخل آن، با یک تأییدیه ی آزمایشی - که داخل Windows Mobile 6 SDK است – امضا می شوند. برنامه ی cabsigntool.exe را در این مسیر هم می توانید پیدا کنید:

\program files\Windows Mobile 6 SDK\Tools\Security

با یک نگاه به دستور های خط فرمانی بالا متوجه می شوید که با این همه دنگ و فنگ اگر قرار باشد هر بار موقع استفاده از cabsigntool.exe این پارامتر ها را وارد نمایید احتمال بروز خطای دستوری هست. پس بهتر آن است که در یک فایل دسته ای bat این دستور ها را قرار دهید و هر بار آن را فراخوانی کنید.

به روز رسانی برنامه ها

بر خلاف برنامه های دسکتاپ که در ویژوال استودیو ۲۰۰۸ تولید می شوند در ویندوز موبایل امکان نصب به روش ClickOnce – که امکان به روز رسانی فایل های از قبل نصب شده را در صورت تغییر آن در نسخه ی جدید می دهد – وجود ندارد. لیکن شما می توانید کارکرد مشابه این را مثلا به وسیله ی وب سرویس ایجاد نمایید. در این مقاله روشی مشابه ClickOnce برای به روز رسانی برنامه های ویندوز موبایل خواهید دید. این مثال صرفا یک نقطه ی آغاز خواهد بود و به شما اطلاعات کافی خواهد داد که خودتان بتوانید به این روش برنامه های تان را به روز کنید.

برای به روز رسانی خودکار برنامه های تان می توانید از وب سرویسی استفاده کنید که آخرین نگارش منتشر شده ی برنامه را اعلام و با نگارش برنامه ی مورد استفاده ی فعلی مقایسه کند. از آن جا که شما باید یک برنامه ی در حال اجرا را به روز رسانی کنید باید یک برنامه ی نصب کننده ی مجزا - همانند نصب کننده ای که از ابتدا برنامه تان را روی دستگاه نصب کرده است- بنویسید.

نصب یک برنامه ی جدید

برای نصب یک برنامه با قابلیت به روز رسانی خودکار باید برنامه تان را از طریق یک وب سایت ASP.NET نصب کنید. برای دانلود برنامه، کاربر باید سایت را برای پیدا کردن محل دانلود آن جستجو کند.

شکل ۳۶: وب سایت حاوی برنامه و دانلود به روز رسانی آن

در شکل ۲ شما یک وب سایت می بینید که یک برنامه ی جدید برای دانلود به صورت آماده دارد. با کلیک روی آن برنامه نصب نخواهد شد بلکه به جای آن یک برنامه ی نصب کننده روی دستگاه نصب می شود. برنامه ی نصب کننده در همان پوشه ای که برنامه ی اصلی در آن نصب خواهد شد قرار می گیرد. برنامه ی نصب کننده در عین حال یک میانبر هم نام با برنامه ی اصلی در پوشه ی Program Files ایجاد خواهد کرد.

پس از دانلود برنامه ی به روز رسانی، کاربر می تواند در پوشه ی Programs روی آن کلیک و برنامه ی به روز رسانی را اجرا نماید. از آنجا که برنامه ی اصلی هنوز قابل دسترسی نیست برنامه ی به روز رسانی به جای آن اجرا می شود (شکل ۳)

شکل ۳۷: اولین اجرای برنامه

برنامه ی به روز رسانی از طریق پارامتر خط فرمان متوجه می شود که آیا توسط کاربر اجرا شده است یا از داخل برنامه ی اصلی. حالت دوم زمانی پیش می اید که به روز رسانی جدیدی داخل وب سایت منتشر شده باشد. کد زیر – که به عنوان بخشی از برنامه ی به روز رسانی داخل رویداد Form_Load اجرا می شود- نشان می دهد که چطور اجرای بار اول را از اجرای حالت به روز رسانی تشخیص دهیم.

```
private void MainForm_Load(object sender, EventArgs e)
{
    if (_args == null || _args.Length == 0 || _args[0].Length == 0)
    {
        _args = new string[1];
        _args[0] = initialURL;
        label1.Text = Properties.Resources.InstallString;
    }
    else
    {
        label1.Text = Properties.Resources.UpgradeString;
    }
}
```

زمانی که برنامه نصب می شود میانبر داخل پوشه ی Program Files را طوری تغییر می دهد که در اجرای بعدی که کاربر برنامه را از داخل پوشه ی Program Files اجرا می کند برنامه ی اصلی اجرا شود نه برنامه ی به روز رسانی.

شکل ۳۸: نصب برنامه و اجرای آن از طریق برنامه ی به روز رسانی (application updater)

پس از نصب برنامه توسط برنامه ی نصب کننده، برنامه ی اصلی به طور خودکار توسط نصب کننده اش اجرا می شود. این کار به محض پایان برنامه ی نصب توسط کاربر از طریق گزینه ی Menu | Exit انجام می شود. کد زیر نشان می دهد که چطور برنامه ی اصلی را از داخل برنامه ی نصب کننده اجرا کنیم. البته کد زیر همه ی آنچه شما به دنبالش هستید، نیست ولی می توانید به راحتی آن را دستکاری کنید و بسته به نیاز تان کم و زیادش کنید:

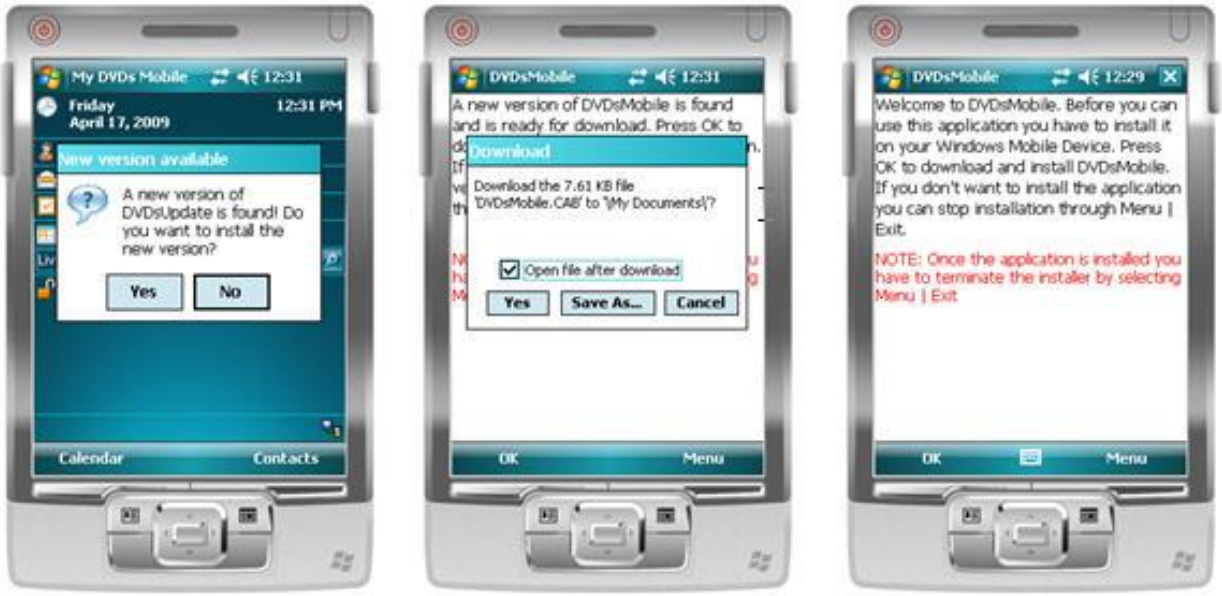
```
private void menuExit_Click(object sender, EventArgs e)
{
   if (label1.Text.Equals(Properties.Resources.InstallString) && ! appInstalled)
   {
      Close();
   }
   else
   {
      StartDVDsMobile();
   }
}

private void StartDVDsMobile()
{
   Process theApp = null;
   try
   {
      theApp = Process.Start(@"\Program Files\DVDsMobile\DVDsMobile.exe",
      Process.GetCurrentProcess().Id.ToString());
   }
   catch (Win32Exception exc)
   {
      if (exc.NativeErrorCode == ERROR_FILE_NOT_FOUND)
      {
```

```
            MessageBox.Show("DVDsMobile.exe not found.");
        }
        else if (exc.NativeErrorCode == ERROR_ACCESS_DENIED)
        {
            MessageBox.Show("No permissions to start DVDsMobile.exe.");
        }
    }
}
```

در قطعه کد بالا مدیر رویداد menuExit_Click را می بینید که تعیین می کند آیا برنامه ی اصلی را اجرا نماید یا فقط از برنامه ی نصب کننده خارج شود. اگر نگارش جدیدتر برنامه نصب شده باشد، برنامه ی به روز رسانی، نگارش جدید را اجرا می کند و شناسه ی پردازش یا همان process identification خود را به داخل آن ارسال می کند که برنامه ی اصلی بتواند برنامه ی به روز رسانی را از حافظه خارج کند. با این که در ظاهر این کار ضروری به نظر نمی رسد اما از نظر این که توجه کاربر را به اجرای برنامه ی جدید پیش از اتمام برنامه ی فعلی جلب می کند مفید است.

به روز رسانی برنامه

هر بار که برنامه توسط کاربر اجرا می شود، می تواند یک وب سرویس از همان وب سایتی که برنامه ی نصب کننده را از آن دانلود کردیم، فراخوانی نماید. وب سرویس می تواند شماره ی آخرین نگارش تمامی اسمبلی ها مورد استفاده برنامه را اعلام نماید تا برنامه ی اصلی آن ها را با شماره ی نگارش های خودش مقایسه کند. اگر نگارش های جدید در وب سایت منتشر شده باشد برناه یک اعلان به کاربر نشان می دهد که آیا می خواهد برنامه را به روز نماید یا این که نگارش فعلی را اجرا نماید. این کار باعث می شود که آغاز اجرای برنامه کمی طولانی تر شود ولی مزیت این را هم دارد که کاربر همیشه از آخرین نگارش استفاده می کند. برای ساده تر شدن کار، برنامه ی ما وصل بودن یا نبودن شبکه را بررسی نمی کند. البته در برنامه ی واقعی می توانید با استفاده از امکانات Stat & Notification Broker این امکان را فراهم نمایید.



شکل ۳۹: به روز رسانی خودکار در عمل

برای دانلود نگارش جدید برنامه، برنامه ی به روز رسانی از طریق یک کنترل مرورگر وب (Web Browser) مخفی وارد سایت می شود و برنامه را دانلود می کند. استفاده از کنترل مرورگر وب، کار دانلود برنامه را بدون نیاز به کار اضافی با وب سرویس امکان پذیر می نماید. کد مربوط به وب سایت را در این مقاله اشاره ای نکرده ام ولی شما می توانید یک روند کامل از این کار را در بخش هفتم همین سری مقاله ببینید. در کد زیر می بینید که چطور برنامه ی اصلی بررسی می کند که آیا نگارش جدیدترش منتشر شده است یا نه:

```
private void MainFormNoTouch_Load(object sender, EventArgs e)
{
   this.Text = Properties.ResourcesNoTouch.MainFormNoTouchTitle;
   this.menuItemExit.Text = Properties.ResourcesNoTouch.MenuItemExitText;
   if (_processID != -1)
   {
      // We have just been updated, so kill our update process.
      Process.GetProcessById(_processID).CloseMainWindow();
   }
   else
   {
      DVDsMobileUpdateService.DVDsMobileUpdateService updateService =
        new DVDsMobileNoTouch.DVDsMobileUpdateService.DVDsMobileUpdateService();
      string[] latestDVDsMobileVersion = updateService.DVDsMobileVersion().Split(
        new char[] { '.' });
      string[] latestDVDsMobileVersionNoTouch =
        updateService.DVDsMobileVersionNoTouch().Split(
        new char[] {'.'});
      string[] currentDVDsMobileVersion = _callingAssembly.Split(
        new char[] { '.' });
      Version currentDVDsMobileVersionNoTouch =
        Assembly.GetExecutingAssembly().GetName().Version;
      bool update = false;
      update = (Convert.ToInt32(latestDVDsMobileVersion[0]) >
                  Convert.ToInt32(currentDVDsMobileVersion[0])) ||
               (Convert.ToInt32(latestDVDsMobileVersion[1]) >
                  Convert.ToInt32(currentDVDsMobileVersion[1])) ||
               (Convert.ToInt32(latestDVDsMobileVersionNoTouch[0]) >
                  currentDVDsMobileVersionNoTouch.Major) ||
               (Convert.ToInt32(latestDVDsMobileVersionNoTouch[1]) >
                  currentDVDsMobileVersionNoTouch.Minor);
      if (update)
      {
         if (MessageBox.Show(
             "A new version of DVDsUpdate is found! " +
             "Do you want to install the new version?",
             "New version available",
             MessageBoxButtons.YesNo,
             MessageBoxIcon.Question,
             MessageBoxDefaultButton.Button2) == DialogResult.Yes)
         {
             Process.Start(@"\Program Files\DVDsMobile\DVDsMobileAutoUpdater.exe",
               @"http://192.168.2.102:5746/DownloadForm.aspx");
             Application.Exit();
         }
      }
   }
}
```

در این قطعه کد مدیر رویداد Form_Load را برای بررسی این که نگارش جدیدتر منتشر شده است یا نه، استفاده می کنیم. بدین منظور دو متد مربوط به وب سرویس DVDsMocileUpdateService فرا خوانی می شوند. در عین حال برنامه شماره ی نگارش خودش را هم می خواند و اگر نگارش موجود در سایت جدیدتر باشد، یک پنجره ی پیغام برای کاربر نمایش داده می شود که از میان دو گزینه ی به روز رسانی و اجرای نگارش فعلی یکی انتخاب نماید. اگر کاربر گزینه ی به روز رسانی را انتخاب نماید، برنامه ی به روز رسانی اجرا می شود و برنامه ی اصلی، خودش را از حافظه خارج می کند. پس از نصب نگارش جدید روی دستگاه، برنامه ی به روز رسانی، نگارش جدید برنامه ی اصلی را اجرا می نماید.

فصل هفتم

توسعه برای وب موبایل

مقدمه

زمانی که برای ویندوز موبایل برنامه می نویسید یک پرسش را همان اول باید پاسخ دهید. آیا برنامه به صورت محلی روی دستگاه اجرا خواهد شد؟ با پاسخ به این پرسش مشخص خواهد شد که آیا نرم افزار تحت وب مناسب تر هست یا نه. اگر دستگاه مورد نظر فاقد اتصال مطمئن به اینترنت است، نرم افزار تحت وب اصلا به درد نخواهد خورد. به عبارت بهتر اگر دستگاه مورد نظر همیشه می تواند به شبکه ی اینترنت متصل شود، نرم افزار تحت وب، انتخاب بهتری است. یک امتیاز نرم افزار تحت وب این است که نیازی ندارد حتما روی دستگاه ویندوز موبایل اجرا شود و از طرفی هم روی طیف وسیعی از دستگاه ها قابل اجرا است. دستگاه های ویندوز موبایل جدید همگی از نرم افزار های تحت وب پشتیبانی می کنند و این را مدیون IEMobile هستند که بر پایه ی برخی قابلیت های اینترنت اکسپلورر ساخته شده است. در دستگاه های اخیر ویندوز موبایل، کاربران درست همانند دستگاه های دسکتاپ از اینترنت استفاده می کنند. تنها تفاوت فقط اندازه ی کوچک تر صفحه ی نمایش است. در ضمن از امکانات ASP.NET 3.5 و AJAX برای کمتر شدن تبادل داده بین کلاینت (دستگاه) و سرور استفاده می نمایند. این امکان به ویژه در اتصال اینترنت با سرعت کم مانند GPRS بسیار کار راه انداز است.

در این مقاله اطلاعاتی را در باره ی توسعه ی نرم افزار های تحت وب موبایل، تنظیم برنامه ی وب برای پشتیبانی از AJAX و در نهایت استفاده از کنترل های مرورگر داخل پروژه ی Smart Device ویژوال استودیو خواهید دید.

نرم افزار های تحت وب و پشتیبانی از ویندوز موبایل

دستگاه های ویندوز موبایل از همان اول مرورگر وب داشتند ولی تا پیش از ارایه ی ویندوز موبایل ۶، امکانات مرورگر مذکور بسیار محدود و بر مبنای نگارش های قدیمی تر اینترنت اکسپلورر بود. آخرین نگارش (در زمان تألیف این مقاله) مرورگر ویندوز موبایل با نام Internet Explorer Mobile 6 یک مرورگر با امکانات کامل است که کیفیت بالای کار با مرورگر وب در دستگاه های دسکتاپ را برای کاربر به ارمغان می آورد. این مرورگر بهترین سازگاری را با تمامی نگارش های مرورگر های ویندوز موبایل دارد. امکانات جدید و پیشرفته ی آن به کاربران کمک می کند که کار شان را به سرعت انجام دهند که در زیر به آن ها اشاره می کنیم:

پشتیبانی از Jscript نگارش ۵.۷ اینترنت اکسپلورر ۸ که به توسعه دهنده گان امکان می دهد از قابلیت AJAX همانند دسکتاپ در ویندوز موبایل هم استفاده نمایند.

اینترنت اکسپلورر موبایل ۶ امکان نمایش هر دو حالت ویژه ی موبایل و معمولی وب سایت ها را دارد.

اینترنت اکسپلورر موبایل ۶ بسته به تنظیم کاربر می تواند به صورت مرورگر دسکتاپ یا مرورگر موبایل عمل کند.

کاربران می توانند از امکانات صفحه ی لمسی و همچنین حالت های مختلف بزرگ نمایی (Zoom) استفاده نمایند.

حالت های مختلف نمایش وب سایت

زمانی که کاربر یک وب سایت را باز یا یک برنامه ی تحت وب را اجرا می نماید، مرورگر کلاینت توسط یک متغیر رشته ای UserAgent شناسایی می شود و بسته به مقدار متغر مذکور، نرم افزار تحت وب امکانات بیشتر یا کمتری (مانند پشتیبانی از AJAX) را برای کلاینت فراهم می کند. آخرین نگارش ویندوز موبایل، نگارشی از اینترنت اکسپلورر را استفاده می نماید که به راحتی می تواند به جای مرورگر دسکتاپ یا ویندوز موبایل شناسایی گردد. برای این کار کافی است که از منوی داخل اینترنت اکسپلورر موبایل گزینه ی مربوط به آن را انتخاب کنید.

شکل ۴۰: برنامه ی اینترنت اکسپلورر ۶ موبایل در حالت نمایش دسکتاپ و موبایل

در شکل ۱ نحوه سوئیچ کردن بین حالت های دسکتاپ و موبایل را می بینید. در بخش وسطی شکل ۱ اینترنت اکسپلورر در حالت ویندوز موبایل اجرا می شود که نمایش وب سایت را برای دستگاه ویندوز موبایل بهینه کرده است. در بخش سمت راست شکل ۱ نیز همان وب سایت را این بار در حالت دسکتاپ مشاهده می کنید. با اینکه صفحه ی نمایش کوکتر است لیکن کاربر همان حالت مرورگر ویندوز دسکتاپ را دارد. در یک دستگاه با امکان صفحه ی لمسی جا به جایی در صفحه ی وب خیلی راحت تر انجام می شود.

تشخیص قابلیت های مرورگر

برای تشخیص امکانات مرورگر کلاینت می توانید از یک متغیر رشته ای UserAgent استفاده نمایید. مقادیر زیر برای متغیر مذکور صرف نظر از نوع دستگاه (استاندارد یا پروفشنال) برگردانده می شوند:

- Mozilla/4.0 (compatible; MSIE 6.0; Windows CE; IEMobile 8.12; MSIEMobile 6.0)
- Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)

که اولی برای حالت موبایل و دومی برای حالت دسکتاپ است.

اگر مرورگر در حالت دسکتاپ اجرا شود به طور خودکار امکانات AJAX را خواهد داشت. اگر مرورگر در حالت موبایل اجرا شود باید کلاینت به سرور اعلام کند که می تواند در حالت سازگار با ie5 اجرا شود. هر مرورگر روی ویندوز موبایل که نگارش IEMobile آن ۶.۱۲ یا بالاتر باشد این سازگاری را دارد. برای اعلام سازگای می توانید از قطعه کد زیر استفاده نمایید:

```
public static bool IsIEMobileWithAjaxSupport(string input)
{
   const string mobileBrowser = "IEMobile ";
   bool ajaxSupported = false;
   if (input.Contains(mobileBrowser))
   {
      string version = input.Substring(input.IndexOf(mobileBrowser) +
         mobileBrowser.Length);
      version = version.Remove(version.IndexOf(';'));
      string[] versionNr = version.Split(new char[] { '.' });
      int IEMobileMajor = Convert.ToInt32(versionNr[0]);
      int IEMobileMinor = Convert.ToInt32(versionNr[1]);
      ajaxSupported = IEMobileMajor > 6 || (IEMobileMajor == 6 && IEMobileMinor >= 12);
   }
   return ajaxSupported;
}
```

کد ۷: تشخیص امکانات اینترنت اکسپلورر موبایل

اگر نگارش اینترنت اکسپلورر موبایل ۶.۱۲ یا بالاتر باشد می توانیم توسط قطعه کد زیر امکانات مرورگر کلاینت را به سرور اطلاع دهیم. این کار را در بخشی از متد FrameworkInitialized داخل نرم افزار تحت وب انجام می دهیم. توجه داشته باشید که این کار یک بار انجام می شود. هر بار که پیغام postback دریافت شود برای نمونه به هنگام بارگذاری مجدد (refresh) بخشی از صفحه نیازی به مقدار دهی ClientTarget نیست چون سرور از مقدار آن مطلع شده است.

```
protected override void FrameworkInitialize()
{
    base.FrameworkInitialize();
    if (!IsPostBack)
    {
        if (IsIEMobileWithAjaxSupport(Request.UserAgent))
        {
            ClientTarget = "ie5";
        }
    }
}
```

کد ۸: ارسال اطلاعات مرورگر کلاینت به سرور

تشخیص ویندوز موبایل استاندارد از پروفشنال

از آن جا که هر دو نگارش استاندارد و پروفشنال ویندوز موبایل مقدار یکسان برای UserAgent برمی گردانند شما باید از روش دیگری برای تشخیص نوع دستگاه استفاده نمایید. در یک درخواست مرورگر اطلاعات اضافی نیز به سرور ارسال می گردد. یکی از این اطلاعات اندازه ی صفحه بر مبنای پیکسل است. تفاوت اصلی هم در این جا است. بر خلاف برنامه های Smart Client ، نرم افزار های تحت وب برای هر دو نوع استاندارد و پروفشنال ویندوز موبایل از کنترل های یکسانی استفاده می نمایند. دلیلش هم آن است که مرورگر اینترنت در هر دوی آن ها یکسان است. بنا بر این یک نرم افزار تحت وب در دستگاه ویندوز موبایل استاندارد می تواند دارای دکمه (button) باشد. شما به راحتی با دکمه های جابجایی می توانید روی دکمه ی مورد نظر تان رفته، با دکمه ی اکشن روی ان کلیک کنید. فیلد هدر ("UA-Pixels") تعداد واقعی پیکسل ها را بر می گرداند یعنی در دو حالت تفکیک گرافیکی بالا و معمولی مقادیر درست را بر می گرداند.

برای تنظیم مرورگر در حالت نمایش بهینه باید از برچسب MobileOptimized (tag) استفاده نمایید که اندازه ی صفحه ی نمایشی را که نرم افزار طبق آن توسعه داده شده است، مشخص نماید. کد زیر داخل برنامه برچسب MobileOptimized را بسته به عرض صفحه ی نمایش مقدار دهی می کند:

```
protected void Page_Load(object sender, EventArgs e)
{
   if (!IsPostBack)
   {
       if (IsIEMobileWithAjaxSupport(Request.UserAgent))
       {
           if (this.Header != null)
           {
               HtmlMeta mobileOptimized = new HtmlMeta();
               mobileOptimized.Name = "MobileOptimized";
               mobileOptimized.Content =
                  Request.Headers["UA-pixels"].Split(new char[] { 'X', 'x' })[0];
               Header.Controls.Add(mobileOptimized);
           }
       }
   }
}
```

کد ۹: تنظیم عرض سمت راست صفحه برای راحتی کاربر

با استفاده از اطلاعات Request.Header می توانید دستگاه های مختلف را تشخیص داده، حالت نمایش را برای همه نوع دستگاه بهینه کنید.

در شکل ۲ همان صفحه ی وب را می بینید که برای نمایش در دو دستگاه استاندارد و پروفشنال ویندوز موبایل بهینه شده است.

شکل ۴۱: بارگذاری صفحه در اندازه ی فونت متفاوت در دستگاه های مختلف

قطعه کد زیر نشان می دهد که چگونه می توانید اندازه ی قلم کنترل های وب را بسته به دستگاه و به ویژه عرض صفحه ی نمایش دستگاهی که برنامه روی آن اجرا می گردد، تغییر دهید. در این جا باید از امکانات ASP.NET 3.5 ممنون باشید و این که می توانید مشخصات کنترل ها را در فایل کد پس زمینه (code – behind) تغییر دهید.

```
protected void Page_Load(object sender, EventArgs e)
{
    string currentTime = DateTime.Now.ToLongTimeString();
    if (!IsPostBack)
    {
        if (IsIEMobileWithAjaxSupport(Request.UserAgent))
        {
            if (this.Header != null)
            {
                HtmlMeta mobileOptimized = new HtmlMeta();
                mobileOptimized.Name = "MobileOptimized";
                mobileOptimized.Content = Request.Headers["UA-pixels"].Split(
                    new char[] { 'X', 'x' })[0];
                Header.Controls.Add(mobileOptimized);
                pageWidth = Convert.ToInt32(mobileOptimized.Content);
            }
        }
    }
    if (pageWidth < 240)
    {
        Label1.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        Label2.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        Label3.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        Label4.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
```

```
        Label5.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        DropDownList1.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        Button1.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
        Menu1.Font.Size = System.Web.UI.WebControls.FontUnit.Small;
    }
}
```

کد ۱۰: تنظیم اندازه ی فونت متناسب با مدل دستگاه کلاینت

می بینید که اینترنت اکسپلورر موبایل آخرین تکنولوژی های موجود در نگارش دسکتاپ را تیز پشتیبانی می کند که اغلب امکانات ASP.NET 3.5 را در موبایل در اختیار تان قرار داده است.

ترکیب وب و اسمارت کلاینت

اگر یک برنامه ی اسمارت کلاینت (Smart Client) توسعه می دهید باید از یک کنترل مرورگر (Browser Control) استفاده کنید. کنترل مرورگر در آخرین نگارش ویندوز موبایل به شما امکان استفاده ار مرورگر وب در دسترس را می دهد و این یعنی امکان استفاده از ASP.NET 3.5 و امکانات AJAX از داخل یک برنامه ی اسمارت کلاینت. ترکیب وب و اسمارت کلاینت این امکان را می دهد که کاربر حتی بدون اتصال به شبکه نیز برنامه را اجرا نماید. اگر اتصال شبکه برقرار باشد کاربر می تواند از کنترل مرورگر استفاده نماید و مثلا به داده های بانک اطلاعاتی روی سرور دسترسی پیدا کند. این ویژگی مثلا در حالتی که چند کاربر به طور هم زمان روی داده های یکسانی در بانک اطلاعاتی کار می کنند از تداخل پیش گیری می نماید چون این کار کاملا توسط سرور انجام می شود. امتیاز دیگر این روش به هنگام پردازش های سنگین از داخل یک کنترل مرورگر خودش را نشان می دهد مثلا برای دریافت اطلاعات نقشه های زمینی. به جای استفاده از وب سرویس شما می توانید تمام عملیات سروری را داخل یک کنترل مرورگر انجام دهید.

کنترل مرورگر که در برنامه ی اسمارت کلاینت خود از آن استفاده می کنید همان مقدار UserAgent مرورگر مستقل اینترنت اکسپلورر موبایل را بر می گرداند. اما کنترل مرورگر را نمی توان به حالت موبایل معرفی کرد. از آن جا که دستگاه ویندوز موبایل صفحه نمایش کوچکی دارد ممکن است بخواهید صفحات خاصی را برای ترکیب کنترل مرورگر در داخل یک برنامه ی اسمارت کلاینت ایجاد نمایید. کد نمونه ی زیر یک برنامه ی کامل را تشان می دهد که اتصال شبکه اینترنت را تشخیص می دهد و در صورت وصل بودن دستگاه امکان نمایش یک صفحه ی وب حاوی اطلاعات در داخل یک کنترل مرورگر هست. اگر شبکه قطع شود کاربر می تواند آدرس های جدید را وارد نماید ولی نمی تواند صفحات آن ها را مرور نماید. البته این مثال بسیار ساده است و در نمونه ی برنامه ی واقعی امکانات بیشتری می توانید قرار دهید.

```
public partial class Form1 : Form
{
   private SystemState nrNetWorkConnections;
   private SystemState deviceCradled;
   public Form1()
   {
       InitializeComponent();
       nrNetWorkConnections = new SystemState(SystemProperty.ConnectionsCount);
       nrNetWorkConnections.Changed +=
          new ChangeEventHandler(nrNetWorkConnections_Changed);
       deviceCradled = new SystemState(SystemProperty.CradlePresent);
       deviceCradled.Changed += new ChangeEventHandler(deviceCradled_Changed);
   }
   private void Form1_Load(object sender, EventArgs e)
   {
       bool isCradled = (int)deviceCradled.CurrentValue != 0;
       int nrNetworks = (int)nrNetWorkConnections.CurrentValue;
       tbURL.Select(tbURL.Text.Length, 0);
       btnGo.Enabled = isCradled || nrNetworks > 0;
```

```
    }
    void deviceCradled_Changed(object sender, ChangeEventArgs args)
    {
        bool isCradled = (int)args.NewValue != 0;
        int nrNetworks = (int)nrNetWorkConnections.CurrentValue;
        btnGo.Enabled = isCradled || nrNetworks > 0;
    }
    void nrNetWorkConnections_Changed(object sender, ChangeEventArgs args)
    {
        bool isCradled = (int)deviceCradled.CurrentValue != 0;
        int nrNetworks = (int)args.NewValue;
        btnGo.Enabled = isCradled || nrNetworks > 0;
    }
    private void btnGo_Click(object sender, EventArgs e)
    {
        webBrowser1.Navigate(new Uri(tbURL.Text));
    }
}
```

کد ۱۱: یک برنامه ی کامل اسمارت کلاینت که اتصالات شبکه را تشخیص می دهد

ترکیب اسمارت کلاینت همراه با وب در وقع بهترین حالت از دو روش را در اختیار تان قرار می دهد. شما می توانید بهره وری کاربر همراه با توان پردازش را روی سرور قرار دهید. اگر داده ها به طور مرکزی ذخیره شوند می توانید یک کپی محلی از آن ها را (حتی از طریق وب سرویس) دانلود کنید. این روش مسأله ی تداخل را حل می کند و میزان داده های قابل دسترس روی دستگاه می تواند به همان مقدار مورد نیاز کاربر محدود گردد. این کار صرفا برای صرفه جویی فضای حافظه دستگاه نیست بلکه به لحاظ امنیتی نیز مهم است چون باعث می شود داده های حساس کمتری در هر زمان روی دستگاه قابل دسترس باشند.

شکل ۴۲: برنامه ی تحت وب از داخل یک مرورگر و به صورت بخشی از یک برنامه ی اسمارت کلاینت

فهرست تصاویر

شکل ۱: پنجره ی New Smart Device Project ۶
شکل ۲: محیط طراحی فرم ویژوال استودیو ۲۰۰۸ ..... ۸
شکل ۳: نخستین برنامه شما در حال اجرا در شبیه ساز دستگاه ویندوز موبایل ۱۰
شکل ۴: برنامه ی ویندوز موبایل در حال اجرا روی شبیه ساز دستگاه ۱۳
شکل ۵: تنظیم مشخصات شبیه ساز ۱۴
شکل ۶: اتصال شبیه ساز تلفن همراه به شبیه ساز دستگاه ۱۵
شکل ۷: شبیه ساز دستگاه و شبیه ساز تلفن همراه در حال اجرا ۱۶
شکل ۸: دریافت پیامک داخل یک برنامه ۱۷
شکل ۹: مدیر شبیه ساز در حال اجرای یک شبیه ساز دستگاه ۱۸
شکل ۱۰: تنظیمات شبیه ساز دستگاه ۱۹
شکل ۱۱: مدیر امنیت دستگاه در حال اجرا از داخل شبیه ساز دستگاه ۲۱
شکل ۱۲: ویژوال استودیو ۲۰۰۸ در نمای طراحی فرم ۲۶
شکل ۱۳: خطوط راهنما بهترین دستیار شما در ترکیب بندی واسط کاربر ۲۷
شکل ۱۴: ویندوز موبایل ۶ حرفه ای در حالت عمودی، کنترل ها نه ثابت شده اند و نه بندکشی ۲۸
شکل ۱۵: ویندوز موبایل ۶ حرفه ای در حالت افقی، کنترل ها نه ثابت شده اند و نه بندکشی ۲۸
شکل ۱۶: ویندوز موبایل ۶ حرفه ای در حالت افقی، با کنترل های ثابت و بندکشی شده ۲۹
شکل ۱۷: تغییر جهت نمایش در حالت طراحی ۳۰
شکل ۱۸: فرم پایه ی برنامه برای دستگاه ویندوز موبایل استاندارد ۳۱
شکل ۱۹: فرم پایه ی مشتق شده برای ویندوز موبایل حرفه ای ۳۲
شکل ۲۰: کنترل سفارشی برای اعتبار سنجی کاربر ۳۸
شکل ۲۱: کنترل سفارشی داخل یک برنامه ۳۹
شکل ۲۲: کنترل های UserCredentials (اعتبار سنجی کاربر) و NumericTextBox (ورود داده ی عددی) در حالت طراحی و اجرا داخل شبیه ساز ۴۰
شکل ۲۳: نرم افزار قطب نمای الکترونیکی که یک کنترل سفارشی را استفاده می کند ۴۴
شکل ۲۴: دیاگرام کلاس GPSID ۴۵
شکل ۲۵: تغییر مشخصه های کنترل قطب نما (Compass) به هنگام بروز خطا ۴۷
شکل ۲۶: قطب نمای الکترونیکی در حال اجرا و نمایش داده های GPS ۴۹
شکل ۲۷: اشتراک پوشه به هنگام مکان یابی شبیه سازی شده با FakeGPS ۴۹
شکل ۲۸: دیتاست (DataSet) های با نوع داده ۵۴
شکل ۲۹: فرم های تولید شده ی خودکار در محیط طراحی ویژوال استودیو ۵۵
شکل ۳۰: زمان لازم برای بارگزاری و نمایش داده ها و جا به جایی بین رکورد ها ۵۷
شکل ۳۱: SqlCeResultSet های با نوع داده ۵۹
شکل ۳۲: اتصال جعبه متن Product ID به منبع داده ی نادرست ۶۱
شکل ۳۳: افزودن دستی منبع اتصال (binding source) و انتساب آن به Order_DetailsResultSet ۶۲
شکل ۳۴: زمان لازم برای بارگزاری و نمایش داده ها و جا به جایی بین رکورد ها (مقایسه کنید با شکل ۳) ۶۳
شکل ۳۵: برنامه ی Device Security Manager ۷۰
شکل ۳۶: وب سایت حاوی برنامه و دانلود به روز رسانی آن ۷۲
شکل ۳۷: اولین اجرای برنامه ۷۳

شکل ۳۸: نصب برنامه و اجرای آن از طریق برنامه ی به روز رسانی (application updater) ۷۴
شکل ۳۹: به روز رسانی خودکار در عمل ۷۶
شکل ۴۰: برنامه ی اینترنت اکسپلورر ۶ موبایل در حالت نمایش دسکتاپ و موبایل ۸۱
شکل ۴۱: بارگذاری صفحه در اندازه ی فونت متفاوت در دستگاه های مختلف ۸۴
شکل ۴۲: برنامه ی تحت وب از داخل یک مرورگر و به صورت بخشی از یک برنامه ی اسمارت کلاینت ۸۶