عنوان	**توسعه نرم افزار برای سیستم عامل ویندوز موبایل،** **بخش پنجم: مقدمه ای بر SQL Server CE**
عنوان اصلی	**Windows Mobile App Development Part 5: Intro to using SQL Server CE**
کلمات کلیدی	C#, Win Mobile, SQL-Server, SQL-CE, .NET, Dev
مؤلف	**mstruys, dougturn**
مرجع	http://www.codeproject.com
سطح	مبتدی
مترجم	مهدی عبداللهی http://m0911.wordpress.com
تاریخ انتشار	۴ فروردین ۱۳۸۹
تعداد صفحه	۱۳
مطالب مرتبط	توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش اول: ایجاد نخستین برنامه توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش دوم: شبیه ساز دستگاه و مدیریت شبیه ساز توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش سوم: توسعه ی برنامه با WinForm توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش چهارم: کنترل سفارشی و GPS توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش ششم: امنیت دستگاه و نصب نرم افزار توسعه ی نرم افزار برای سیستم عامل ویندوز موبایل، بخش هفتم: توسعه برای وب موبایل
فایل های ضمیمه	

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. این تفاوت به هنگام استفاده از بانک اطلاعاتی هم هست. اغلب برنامه ها از بانک اطلاعاتی استفاده 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 مقاله مطالبی راجع به SQL Server 2005 Compact Edition و روش های مختلف دسترسی به داده های محلی ذخیره شده روی دستگاه خواهید دید. ما دو روش متفاوت دسترسی به داده ها را از لحاظ کارآیی زمان لازم برای اجرا و میزان کد نوشته شده مقایسه خواهیم کرد. در مثال های کدنویسی دو جدول Orders و Order_Details از بانک اطلاعاتی نمونه ی Northwind را استفاده کرده ایم. نگارش مورد استفاده ی بانک 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 کند. در زمان 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 نگارش روی کامپیوتر های کوچک (Tablet PC) اجرا می شود. از زمانی که ابزارهای توسعه روی کامپیوتر های دسکتاپ نصب می شوند این نگارش روی ماشین های خاصی اجرا می شود.
- SQL Server 2005 Compact Edition 3.1: در سال ۲۰۰۶ منتشر شد و تا حد زیادی بر پایه 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 ها هم بدون محدودیت اجرا می گردد. SQL Server CE سازگاری زیادی با ویرایش های مختلف SQL Server دارد و امکانات کامل بانک اطلاعاتی رابطه ای را در مقیاس کوچک فراهم می کند. این نگارش SQL Server CE روی حافظه ی اصلی 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 شد و در عین حال داخل ویژوال استودیو ۲۰۰۸ نیز موجود است. بر مبنای توسعه ی SQL Server CE 3.1 تولید شده است. این نگارش SQL Server CE امکان همسان سازی با سرور های میکروسافت را از طریق سرویس Microsoft Syncronization برای ADO.NET فراهم می سازد.

زیر ساخت SQL Server 2005 Compact Edition

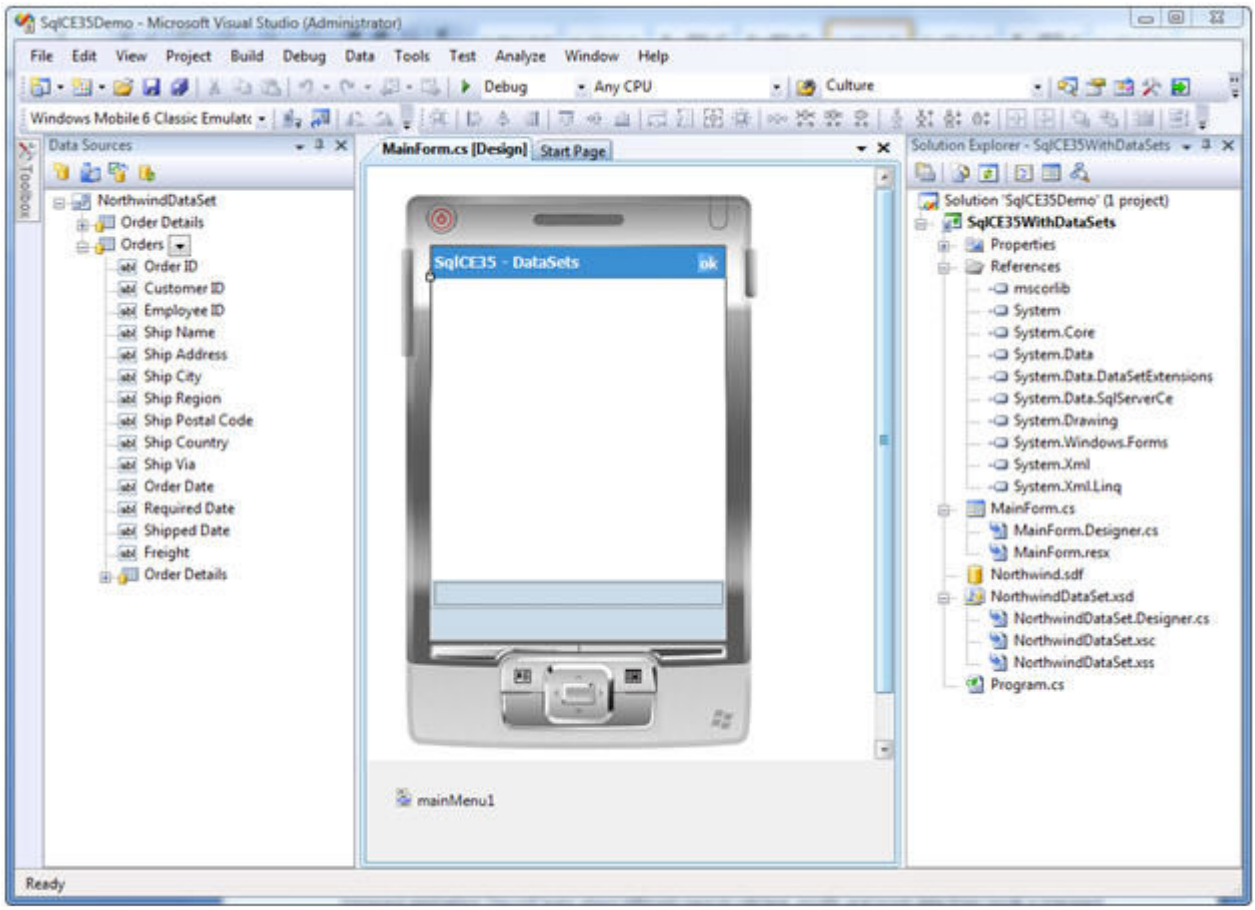
SQL Server CE یک بانک اطلاعاتی سبک است و بر خلاف نگارش های دیگر SQL Server، به صورت یک پردازش همراه با برنامه ای که از آن استفاده می کند، اجرا می شود. این بدان معنی است که SQL Server CE به صورت یک سرور مجزا پیاده سازی نشده است. بانک اطلاعاتی آن در یک فایل منفرد با پسوند sdf ذخیره می شود و همچنین امکان 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 ۱: 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 ۲: 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())
```

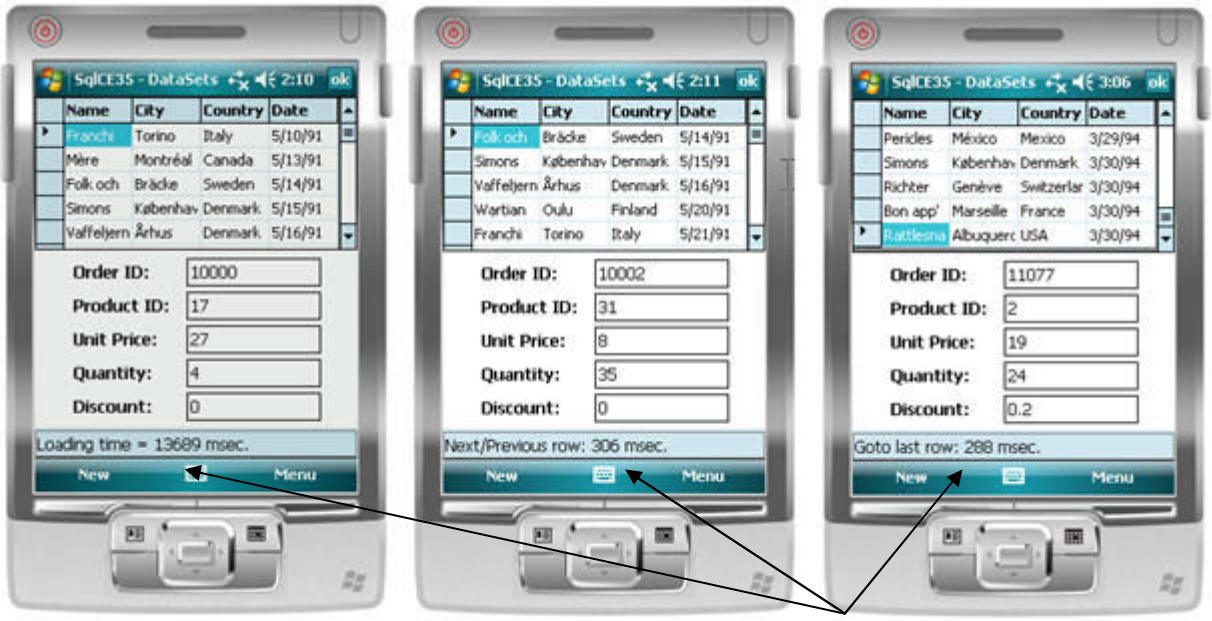
```
        {
            this.ordersTableAdapter.Fill(this.northwindDataSet.Orders);
        }
    }
    private void newMenuItemMenuItem_Click(object sender, EventArgs e)
    {
        ordersBindingSource.AddNew();
        SqlCE35WithDataSets.OrdersEditViewDialog ordersEditViewDialog =
            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 ۳: 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.

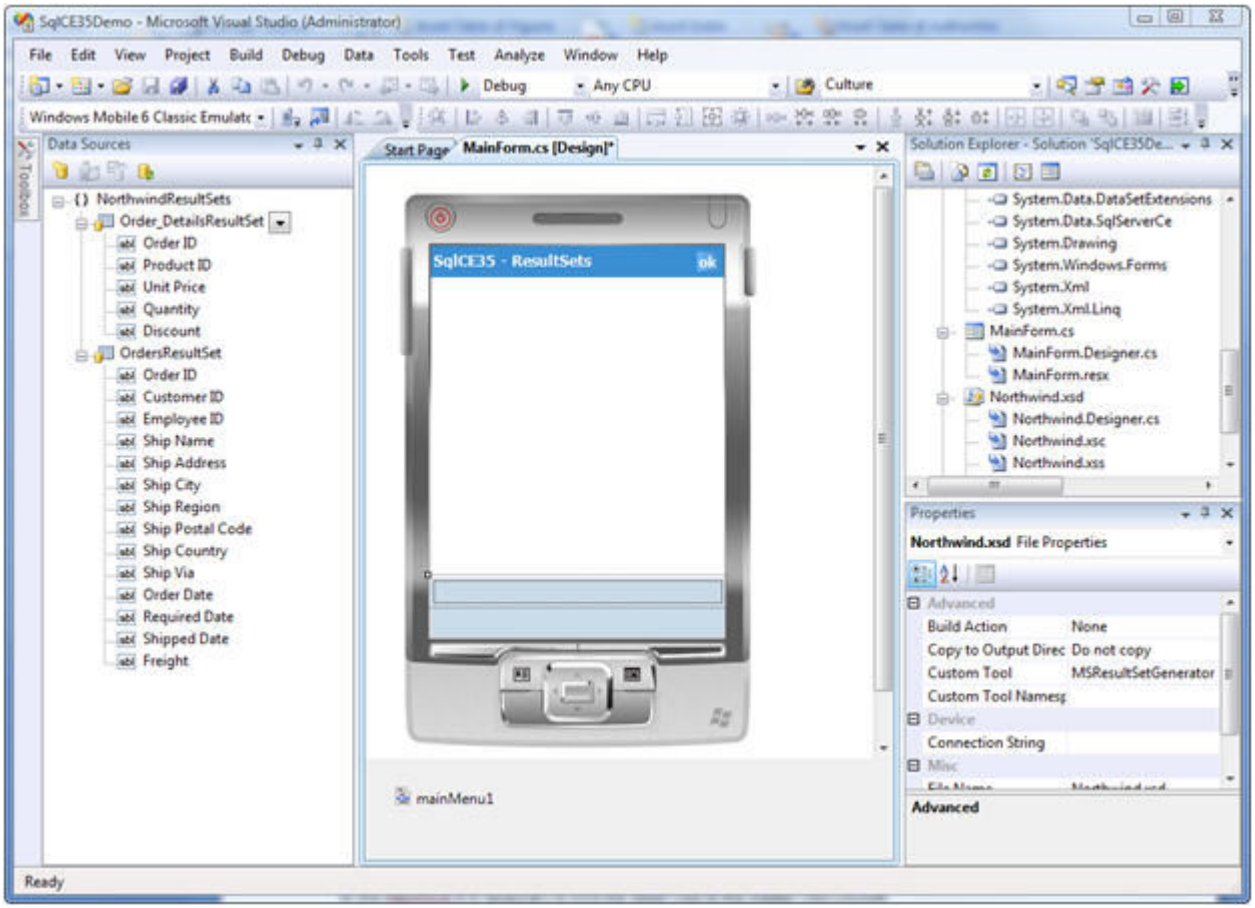
استفاده از SqlCeResultSets با نوع داده

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 ۴: SqlCeResultSet های با نوع داده

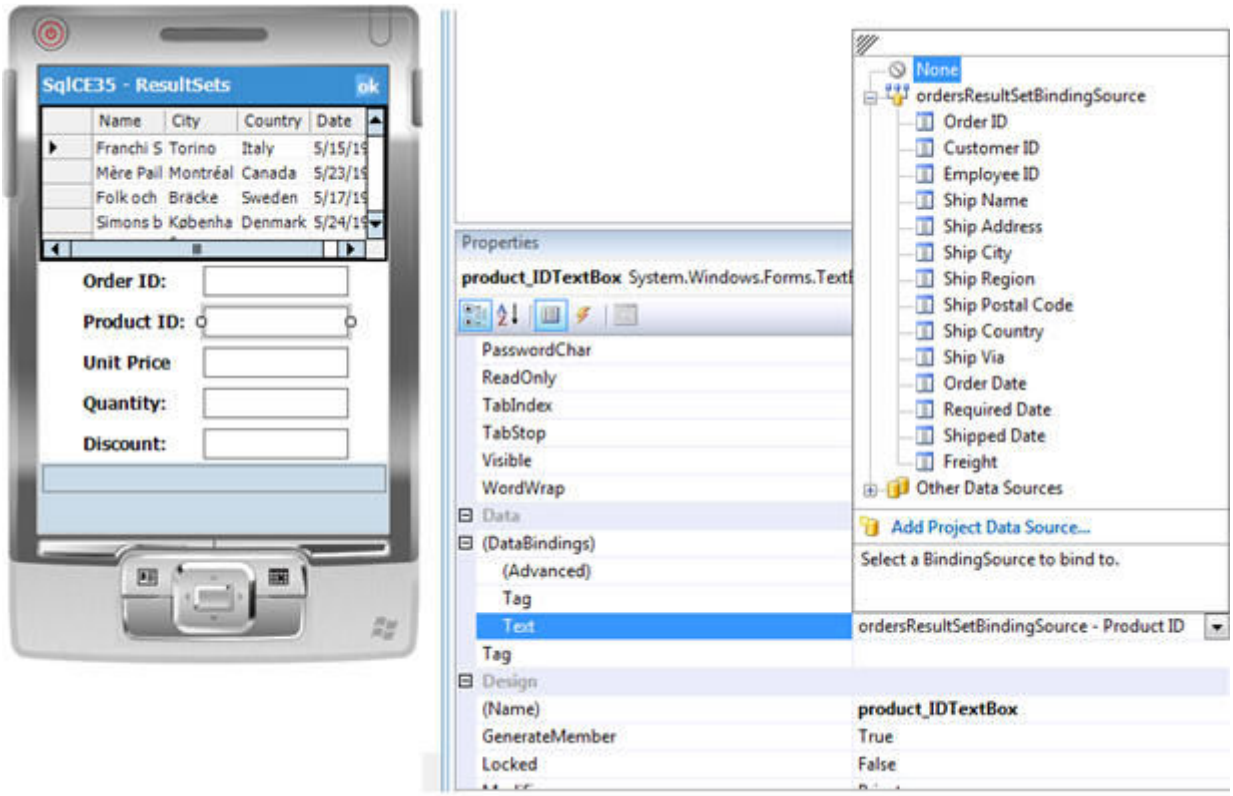
برای نمایش داده های جدول Order_Details هم زمان با رکورد انتخاب شده از جدول Order شما باید اندکی به خودتان زحمت داده، چند خط کد نویسی کنید. یک نقطه ی شروع خوب برای این کار نوشتن کد و بهبود کارکرد Order_DetailsResultSet است. روی Northwind.xsd در سولوشن اکسپلورر راست کلیک کنید. ویژوال استودیو یک فایل کد 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();
            }
        }
    }
}
```

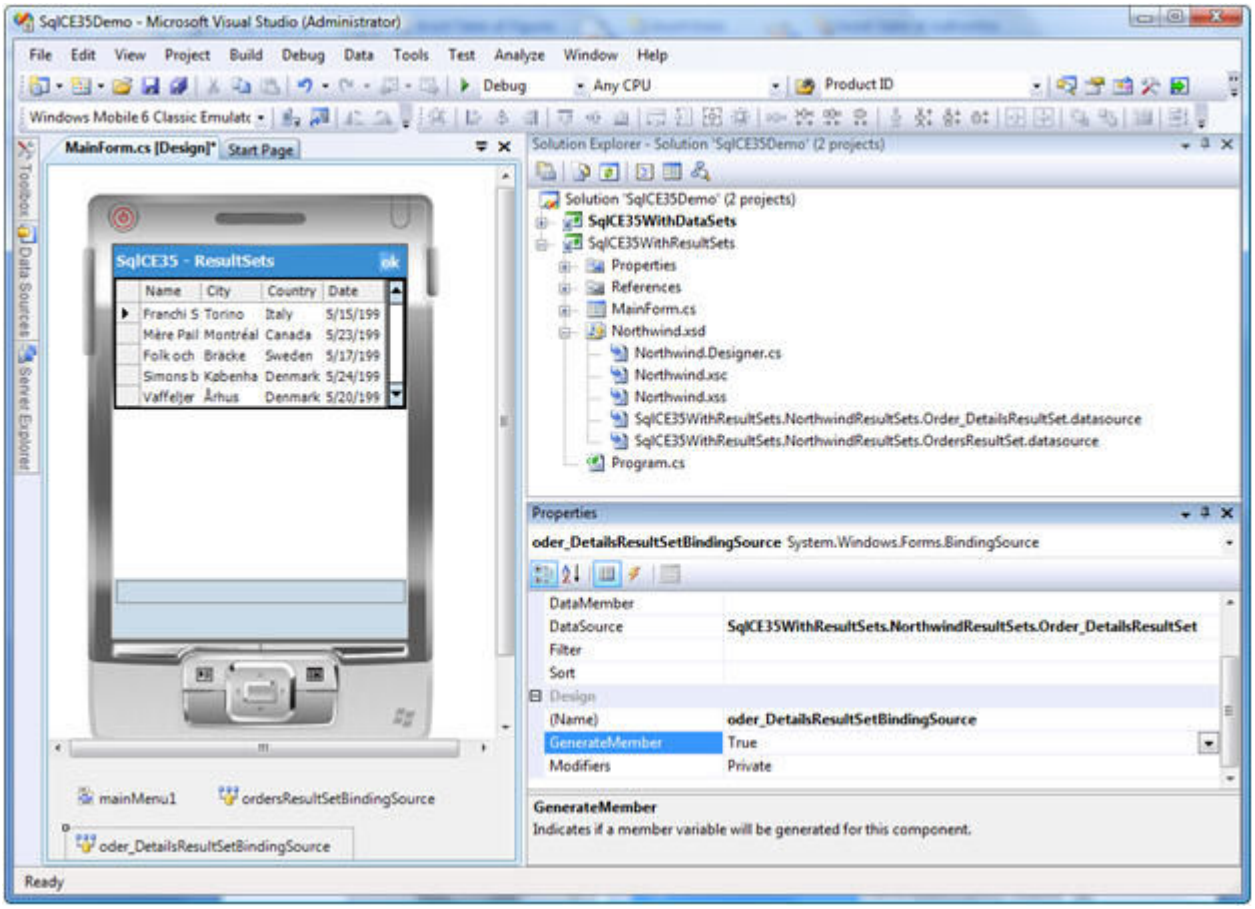
کد ۵: توسعه ی Order_DetailsResultSet

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 ۵: 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 ۶: 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 بـه دسـت آوریـد. سپس با استفاده از آن یک عبارت پرس و جو درست کنید که از جدول Order Details اطلاعات مربوط به جزئیات سفارش انتخـاب شـده را بـه دست آورد. پس از این که یک Order_DetailsResultSet ایجاد و رکورد منطبق با شرایط پرس و جو، در آن ذخیره شـد، بایـد منبـع اتصـال (BindingSource) را به آن متصل کنیم و سپس ResultSet اصلی را از حافظـه خـارج کنـیم. در کـد ۶، روش اتصـال دو جـدول 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 ۷: 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. **(;**