

# بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِیْمِ

یک چند به کودکی به استاد شدیم      یک چند به استادی خود شاد شدیم  
پایان سخن شنو که ما را چه رسید      از خاک برآمدیم و بر باد شدیم

خیام

عنوان :

برنامه نویسی پیشرفته در سی شارپ

جلداول

نگارش :

مسعود نظیفی اصل

بهار ۱۳۸۹

هدف از نگارش این کتاب کمک به کسانی است که علاقه زیادی به برنامه نویسی و سعی در یادگیری آن از مبتدی تا پیشرفته دارند. در ضمن کسانی هم که نمی خواهند برنامه نویسی را یاد بگیرند هم می توانند از این کتاب به عنوان افزایش سطح علمی خود در زمینه کامپیوتر استفاده کنند چون علاوه بر برنامه نویسی مطالبی هم در زمینه های مختلف علوم کامپیوتر گفته شده است. از این کتاب همه افرادی که آشنایی مختصری با کامپیوتر و زبان انگلیسی دارند می توانند استفاده کنند. استفاده از این کتاب به صورت الکترونیکی رایگان است ولی تغییر در مطالب یا برش کتاب و پرینت از صفحات آن بدون اجازه مولف غیر قانونی است.

لطفا انتقادات و پیشنهادات و سوالات خود را به این شماره پیامک کنید :

**Tel : ۰۹۱۴۱۵۰۹۱۰۴**

به دلیل اینکه در بعضی مطالب این کتاب از بعضی سایت های اینترنتی استفاده شده که در حال حاضر یا آدرس آن سایت ها تغییر کرده و یا پاک شده اند دیگر امکان ذکر نام آن سایت ها به عنوان منابع وجود نداشت .

فهرست :

۹.....مقدمه

۱۰.....فصل اول (برنامه نویسی سی شارپ).....

۱۱.....محیط ویژوال سی شارپ.....

۱۴.....جعبه ابزار.....

۱۶.....نحوه برنامه نویسی در سی شارپ.....

۱۸.....دات نت چیست؟.....

۱۹.....متغیرها.....

۲۷.....حلقه ها.....

۲۷.....حلقه for.....

۲۸.....ListBox با استفاده از حلقه for.....

۳۰.....جستجو در ListBox.....

۳۳.....حلقه while.....

۳۴.....حلقه do while.....

۳۴.....شرط هاIF.....

۳۵.....Else if.....

۳۶.....Switch.....

۳۸.....مثال ها.....

۴۰.....	.....Math
۴۱.....	.....توابع
۴۲.....	.....انواع توابع
۴۶.....	.....آرایه ها...
۴۹.....	.....foreach حلقه
۵۰.....	.....جعبه پیغام
۵۱.....	.....آرایه های چند بعدی.....
۵۳.....	.....کلاس ها در سی شارپ.....
۵۷.....	.....نوع های محلی و عمومی.....
۵۹.....	.....ارث بری.....
۶۰.....	.....سازنده کلاس.....
۶۵.....	.....Get set
۶۷.....	.....استفاده از چندین فرم.....
۶۹.....	.....ویژگی فرمها.....
۷۱.....	.....ثابتها.....
۷۲.....	.....Dialogs
۷۹.....	.....Timer
۸۱.....	.....System.Collections

۸۴	.....	کنترل استثناها
۸۸	.....	System.Drawing
۹۰	.....	پردازش تصویر
۹۵	.....	گرافیک
۹۹	.....	DLL
۱۰۴	.....	Threading
۱۰۹	.....	کار با فایلها (فشرده سازی و رمزنگاری)
۱۱۱	.....	نحوه ایجاد فایل
۱۱۴	.....	رمز کردن فایل
۱۱۶	.....	فشرده سازی
۱۱۷	.....	الگوریتم رایندال
۱۲۱	.....	الگوریتم MD5
۱۲۳	.....	اضافه کردن فایل به برنامه اجرایی
۱۲۵	.....	کلاس های system.io
۱۲۸	.....	UserControl
۱۳۲	.....	مدیا
۱۳۵	.....	<b>فصل دوم (پایگاه داده)</b>
۱۳۶	.....	محیط اکسس
۱۳۸	.....	دستورات اکسس

۱۴۱.....	فصل سوم (کلاسهای دیتابیس).....
۱۴۲.....	...System.Data.OleDb
۱۴۳.....	.....OleDbConnection
۱۴۵.....	.....OleDbCommand
۱۴۶.....	.....OleDbDataAdapter
۱۴۷.....	.....DataTable
۱۴۸.....	مثال برای پایگاه داده.....
۱۵۵.....	نحوه گزارش گیری ...
۱۶۶.....	..... LINQ
۱۶۸.....	.....LINQ to SQL
۱۷۵.....	.....sql server پروسیجرها و

**۱۷۹..... فصل چهارم (ASP .NET).....**

۱۸۱.....	.....asp محیط
۱۸۲.....	.....کد نویسی صفحات وب و ابزارها.....
۱۸۹.....	.....کنترل های html
۱۹۱.....	.....Table
۱۹۴.....	.....Validation
۲۰۴.....	.....MasterPage

۲۰۹.....Navigation

۲۱۲.....SiteMapPath

۲۱۳.....FileUpload

۲۱۵.....AJAX

۲۱۷.....(ObjectDataSource)SQL کار با پایگاه داده

۲۳۵.....کوکی ها

۲۳۷.....مراجع



مقدمه :

شاید این سوال برای شما پیش آید که هدف از برنامه نویسی چیست در کل تعریفی که برای آن ارایه می شود یکی نیست بنابراین به چند مورد از آن اشاره می کنم .

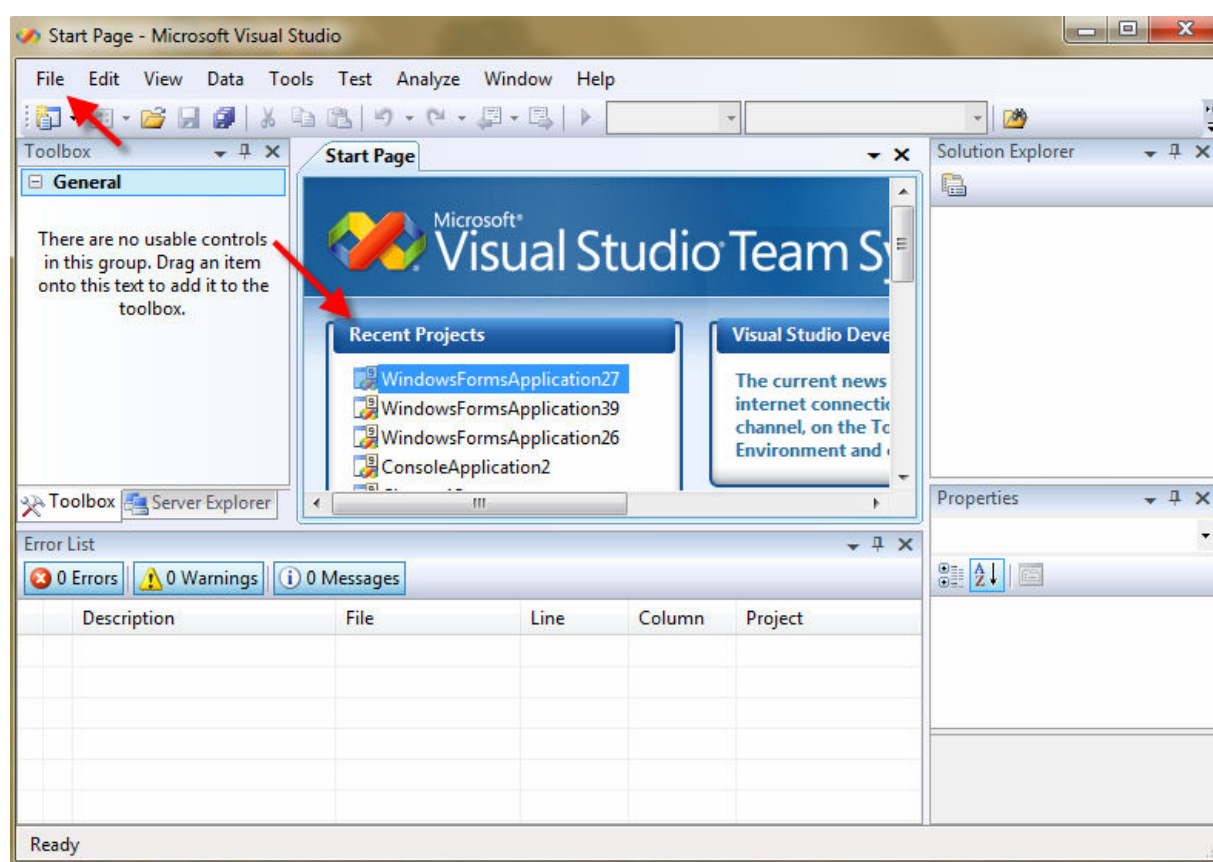
برنامه نویسی یعنی اینکه ما بتوانیم خواسته های خودمان را بر روی کامپیوتر اجرا کنیم و جوابهایی که می خواهیم به صورت دقیق و سریع به دست آوریم مثلا برای محاسبه توابع ریاضی که در ذهنمان است و می خواهیم به جواب برسیم کافیت آن را بر روی کامپیوتر پیاده سازی کنیم ولی اینکار به همین راحتی نیست بلکه ما باید بتوانیم زبان کامپیوتر را نیز بفهمیم دانستن زبان کامپیوتر متکی به دانستن نحوه عملکرد سخت افزار است یعنی اینکه اعضای سخت افزاری کامپیوتر مثل Cpu و Ram چگونه کار می کنند و ارتباط آنها به چه صورت است مثلا کدام پایه های cpu به عنوان ورودی یا خروجی (اطلاعات) یا کنترل (بخش های دیگر) عمل می کنند ولی انجام این کارها به همین سادگی نیستند بنابراین برای راحتی کار واسطی میان انسان و سخت افزار با نام نرم افزار به وجود آمد که زبان ما را به زبان کامپیوتر ترجمه می کنند. برای نوشتن این نرم افزارها هم زبان های برنامه نویسی به وجود آمد که با استفاده از آن می توانیم برنامه های خودمان را به راحتی بنویسیم و توسط کامپیوتر اجرا کنیم از جمله زبان های برنامه نویسی می توان به C, C++, C#, VB اشاره کرد که بعضی از آنها در نحوه نوشتن برنامه شبیه هم هستند در این کتاب ما به بررسی زبان پیشرفته و حرفه ای سی شارپ (C#) می پردازیم و این زبان را از مبتدی تا پیشرفته بررسی می کنیم و به بررسی مطالب جالبی حول این زبان می پردازیم مثل پردازش تصویر کار با فایلها و پایگاه داده. (البته به طور مختصر با اعضای سخت افزار کامپیوتر آشنا خواهیم شد تا بهتر بتوانیم به صورت مفهومی با برنامه نویسی آشنا بشیم).

mSP soft

فصل اول:  
ويژوال سي شارپ

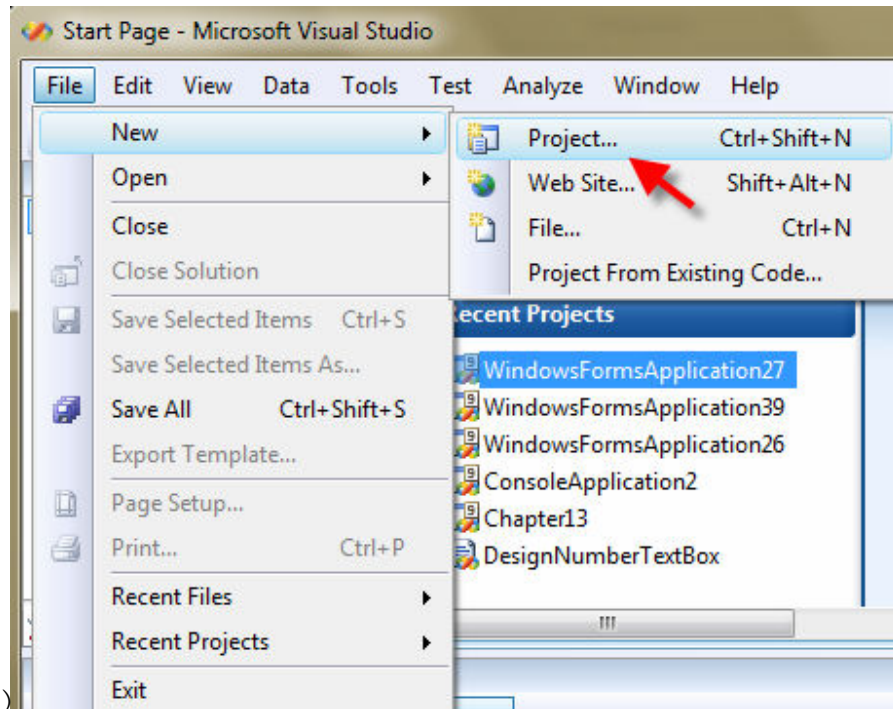
## ۱-۱ محیط ویژوال سی شارپ

ویژوال سی شارپ دارای محیط زیبا و آسانی برای برنامه نویسی است وقتی برای اولین بار برنامه ویژوال سی شارپ را اجرا می کنید پنجره ( شکل ۱-۱) را مشاهده می کنید. در قسمت Recent Project برنامه هایی (پروژه) را که شما قبلا کار کردید را می بینید با کلیک بر روی هر کدام از برنامه ها می توانید آنها را اجرا کنید و اگر خواستید پروژه جدیدی برای خود ایجاد کنید و یا پروژه ای را که در جایی ذخیره کردید را ببینید (که در لیست موجود نیست) می توانید از قسمت File این کار را انجام دهید.



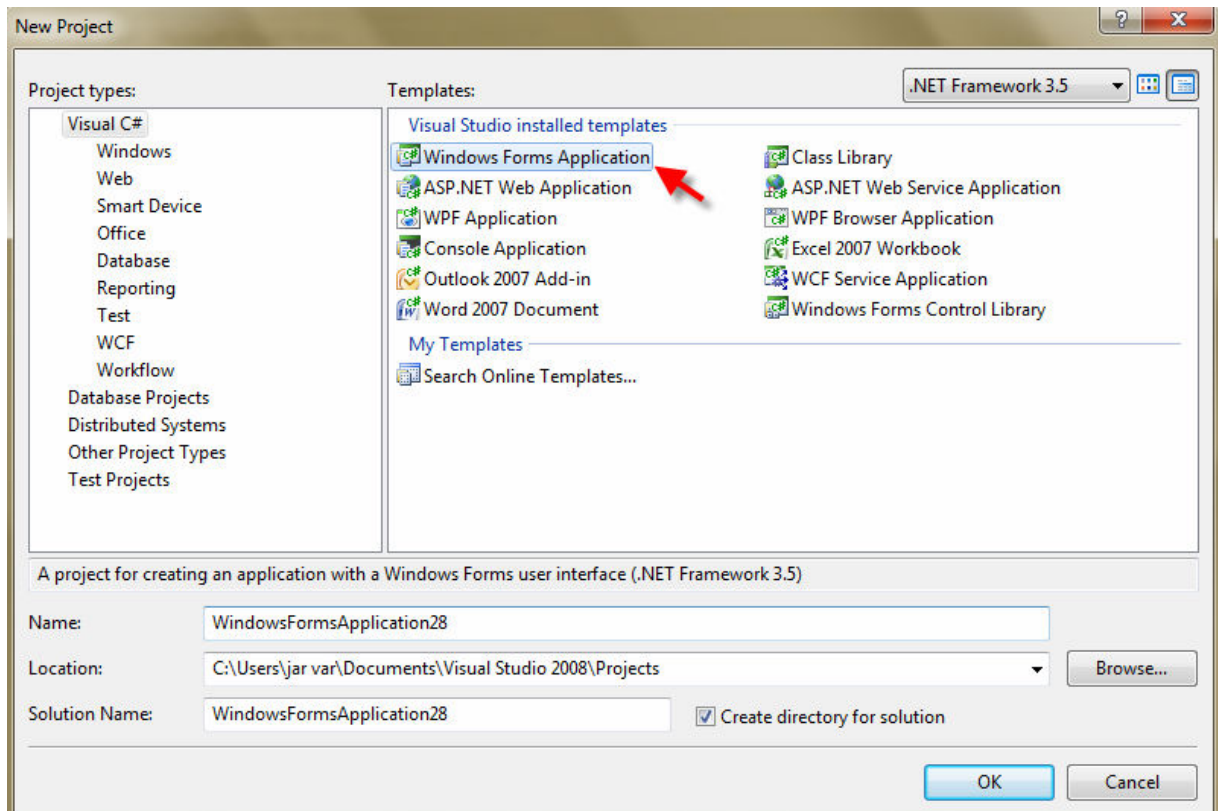
(شکل ۱-۱)

برای ایجاد یک پروژه جدید می توانید از قسمت File و New و Project را انتخاب کنید (شکل ۱-۲)



(۲-۱)

در این صفحه (شکل ۱-۳) بر روی Windows Form Application کلیک کنید تا پروژه جدید ایجاد شود

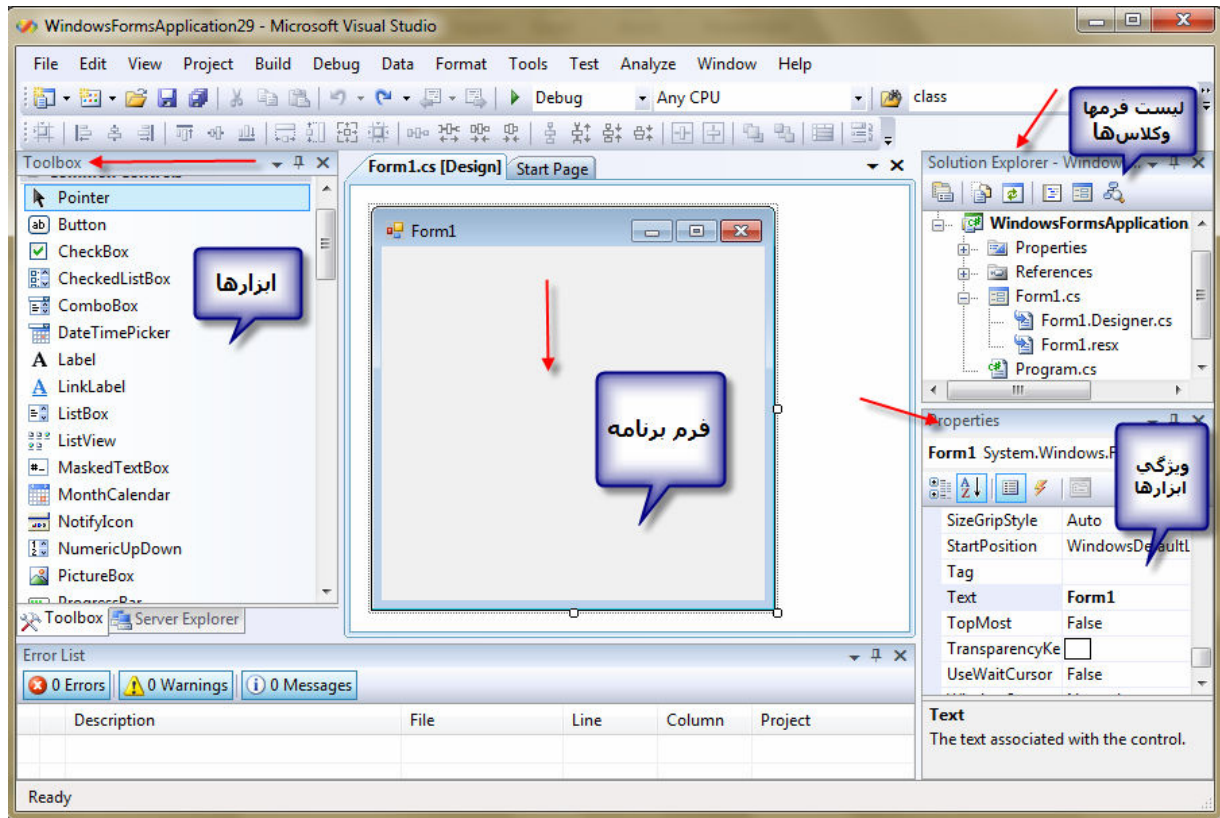


در مورد دیگر آیتم های این صفحه در قسمت های بعدی اشاره خواهیم کرد ولی درباره Windows Form Application و ConsoleApplication این توضیح را بدم که در اولی برنامه هایی که می نویسیم دارای محیط گرافیکی است و کاربر بهتر می تواند از آن استفاده کند ولی در دومی برنامه هایی که می نویسیم در محیط داس اجرا می شوند که امکانات گرافیکی در آن گنجانده نشده است در زیر شکل دو برنامه ماشین حساب در دو محیط را آورده ایم و خواستم تفاوت گرافیکی و راحتی کار با آن ها را مشاهده کنید.



البته باید به این نکته اشاره کنم که سرعت اجرای برنامه در محیط دوم بیشتر از اولی است.

حالا در همان صفحه (شکل ۱-۳) بر روی Windows Form Application کلیک کنید و اگر خواستید برای برنامه خود نامی را انتخاب کنید و مسیری که برنامه در آنجا ایجاد می شود را تغییر دهید بعد بر روی دکمه ok کلیک تا برنامه جدیدی ایجاد شود. در این کتاب بیشتر با Windows Form Application کار خواهیم کرد ولی نحوه کد نویسی در دو محیط یکی است. وقتی برنامه جدیدی ایجاد کردید وارد محیط جدید و اصلی می شوید (شکل ۱-۴) در این محیط فرم اصلی برنامه و ابزارهایی را که قرار است از آنها استفاده کنیم قرار دارد.



(۴-۱)

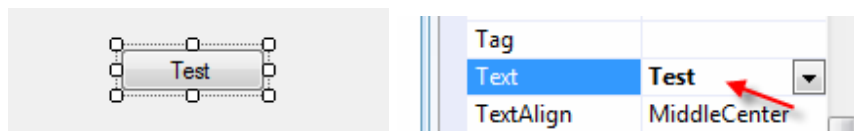
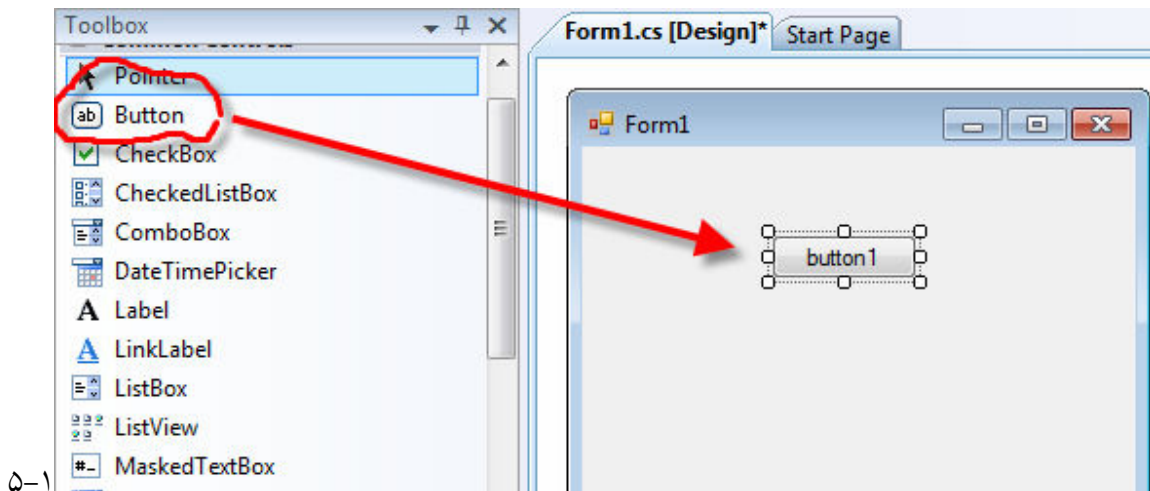
در قسمت چپ جعبه ابزارها قرار دارد البته اگر جعبه ابزارها را مشاهده نکردید می توانید با کلیک بر روی



Toolbox در قسمت بالا

این پنجره یا هر پنجره دیگر را نمایش دهید. در جعبه ابزارها کنترل های مختلفی را مشاهده می کنید که هرکدام ویژگی منحصر به فردی برای خود دارند هر کدام از این ابزارها را در قسمت های بعدی توضیح خواهیم داد. در وسط هم فرم اصلی برنامه قرار دارد که ابزارها بر روی این فرم قرار می گیرد برای قرار دادن ابزارها بر روی فرم کافیست آنها را بکشید و بر روی فرم قرار دهید مثلا برای قرار دادن یک دکمه به همین ترتیب عمل کنید وقتی ابزاری را در صفحه قرار دادید می توانید آنها را جابجا کرده و در مکان دلخواه قرار دهید همچنین می توانید بر روی ابزار کلیک راست کرده و از قسمت **properties**

ویژگی ها آن ابزار را مشاهده کرده و تغییر دهید مثلا در قسمت text می توانید متن دکمه را تغییر دهید .



یا رنگ آن را از قسمت bgcolor و فونت آن را از قسمت Font تغییر دهید

برای آشنایی بیشتر با ابزارها بهتر است هرکدام را به ترتیب بر روی فرم قرار داده و با تغییر ویژگی هایشان با آنها آشنا شویم اولین ابزاری که می خواهیم با آن آشنا شویم همین ابزار Button است (شکل ۵-۱) شما با این ابزار بیشتر آشنایی دارید مثلا در محیط ویندوز وقتی با ماشین حساب کار می کنید دکمه های اعداد را که فشار می دهید همین ابزار است که در ظاهر این عمل را انجام می دهد. ولی کدهای پشت سر این دکمه ها با توجه به وظیفه آن می تواند متفاوت باشد.

Label از این ابزار بیشتر برای توضیحات برنامه استفاده می شود که می تواند در کنار ابزارهای دیگر قرار گیرند.

pictureBox از این ابزار برای نمایش تصاویر در برنامه و ایجاد گراف استفاده می شود.

**textBox** از این ابزار برای وارد کردن و خواندن اطلاعاتی که لازم است استفاده می شود مثلا برای وارد کردن نام و اعداد .

**checkBox** از این ابزار برای انتخاب گزینه های پیش نهادی به کاربر استفاده می شود که کاربر می تواند آنها را انتخاب کند مانند شکل زیر



**monthCalendar** از این ابزار برای نمایش و انتخاب تاریخ استفاده می شود

فوريه, ۲۰۱۰						
شنبه	يكشنبه	دوشنبه	سه شنبه	چهارشنبه	پنجشنبه	جمعه
۳۰	۳۱	۱	۲	۳	۴	۵
۶	۷	۸	۹	۱۰	۱۱	۱۲
۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹
۲۰	۲۱	۲۲	۲۳	۲۴	۲۵	۲۶
۲۷	۲۸	۱	۲	۳	۴	۵
۶	۷	۸	۹	۱۰	۱۱	۱۲

Today: ۲۰۱۰/۱۶/۰۲

**Timer**: یکی از کنترل های جالب تایمر است که به وسیله این کنترل می توانید زمان اجرای برنامه را تعیین کنید مثلا اگر بخواهید یک برنامه در هر دقیقه یک بار اجرا شود از این ابزار استفاده می شود البته باید به این نکته اشاره کنیم بعضی از ابزارها دارای گرافیک مثل **textBox** نیستند بلکه در قسمت کد نویسی با آن کار می شود مثل همین کنترل تایمر.

اینها ابزارهای مهم بودند که به بعضی از آنها اشاره کردیم و به توضیح بقیه آنها در مثالها خواهیم پرداخت و نیز به نحوه استفاده از آنها در محیط برنامه نویسی و کد نویسی می پردازیم.

## ۲-۱ نحوه برنامه نویسی در سی شارپ



مطالبی که در قسمت قبلی مشاهده کردید شامل لایه فیزیکی و ویژوال سی شارپ بود ولی برای نوشتن کد، سی شارپ دارای محیط دیگری برای برنامه نویسی است که برای دسترسی به این محیط می توانید روی فرم برنامه یا هر کلاس دیگر کلیک راست کرده و **View code** را بزنید (شکل ۶-۱) با انجام اینکار وارد محیط برنامه نویسی شکل ۷-۱ می شوید در این قسمت می توانید شروع به برنامه نویسی نمایید. [2]

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication63
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}

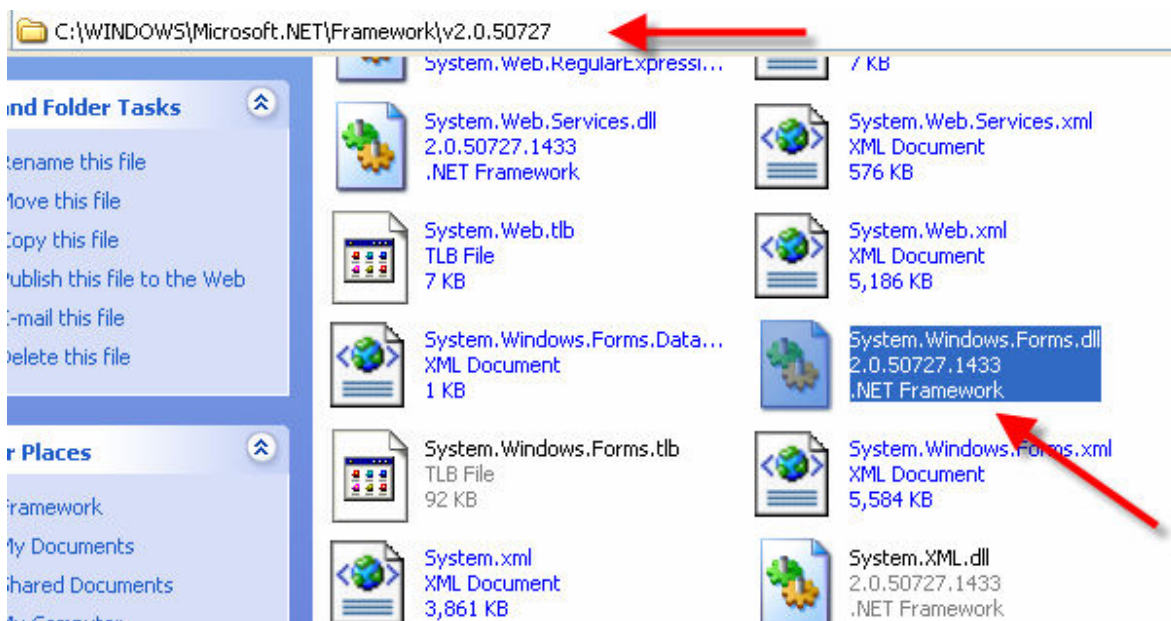
```

شکل ۶-۱

شکل ۷-۱ هدرهایی که به صورت پیش فرض اضافه شده

همانطور که در شکل ۷-۱ مشاهده می کنید یک تعداد کدهایی به صورت پیش فرض به برنامه اضافه شده است در قسمت بالا هدرهای برنامه را مشاهده میکنید که به برنامه اضافه شده است در این هدرها کلاس هایی وجود دارد که ما در برنامه از آنها استفاده می کنیم البته باید به این نکته نیز اشاره کنم که همه هدرها اضافه نشده اند بلکه هدرهایی که عموماً از آنها بیشتر استفاده میشود به برنامه اضافه شده است. این هدرها یا همان کتابخانه های سی شارپ که به برنامه اضافه شده داخل فایل **dll** ذخیره می شوند در شکل ۸-۱ نمونه ای از این کتابخانه

ها را مشاهده میکنید که در قسمت های بعدی به طور مفصل درباره این کتابخانه ها کلاس های داخلی آنها بحث خواهیم کرد.



شکل ۸-۱

همانطور که در شکل ۸-۱ مشاهده می کنید این کتابخانه ها و کتابخانه های دیگر داخل **net framework** قرار میگیرند که می توانید به آنها دسترسی داشته باشید. [1] این کتابخانه ها توسط شرکت سازنده ویژوال استدیو برای راحتی کار در محیط برنامه نویسی ساخته شده اند البته در بخش های دیگر نحوه ساختن این کتابخانه ها توضیح داده شده است این که چگونه بتوانیم برای خودمان یک فایل **dll** بسازیم و در برنامه مون از آن استفاده کنیم.

## Net framework

دات نت چیست ؟

دات نت تکنولوژی است که برنامه هایی را که توسط **visual studio** نوشته می شوند را اجرا می کند و کتابخانه های مورد نیاز را در اختیار برنامه قرار می دهد بنابراین اگر دات نت بر روی کامپیوتر شما نصب نباشد

برنامه های شما اجرا نخواهند شد از ویژگی های دانت این است که برنامه شما دیگر وابسته به سیستم عامل نمی شود مثلا شاید برای شما پیش آمده است که برنامه ای در یک سیستم عامل مثل XP اجرا می شود و در سیستم عامل Vista یا Win7 اجرا نمیشود و برعکس علت این امر اینست که برنامه شما را برای یک سیستم عامل خاص و با ویژگی آن سیستم عامل نوشته شده است و آن سیستم عامل همان برنامه را به طور کامل اجرا می کند و نیازهای برنامه را سیستم عامل در اختیار برنامه قرار می دهد و بنابراین اگر سیستم عامل تغییر کرد دیگر آن برنامه نمی تواند اجرا شود ولی اگر توسط تکنولوژی دانت نوشته شود دیگر این مشکل ها وجود نخواهد داشت. آزمایشهای دانت این است که فایل اجرایی نوشته شده توسط این برنامه دارای ظرفیت بسیار پایین است چون کتابخانه های مورد نیازش را دانت در اختیار برنامه قرار می دهد.

دانت در سیستم عامل های جدید به طور اتوماتیک همراه با نصب سیستم عامل نصب می شود. حال برگردیم به محیط برنامه نویسی تا ببینیم چگونه می توانیم در این محیط برنامه بنویسیم کنیم برای نوشتن برنامه ابتدا باید با دستورات و زبان برنامه نویسی در سی شارپ آشنا شویم. کسانی که با زبان های برنامه نویسی C, C++ آشنایی دارند یادگیری برنامه نویسی در سی شارپ برای آنها آسان است چون بیشتر دستورات شبیه آنهاست.

### ۱-۳ متغیرها:

برای تعریف اعداد صحیح و اعشاری در سی شارپ مانند شکل ۱-۹ عمل می کنیم به این نکته نیز باید اشاره کنم که همه دستورات در سی شارپ به (;) ختم می شود اگر این نشانه را نگذاریم برنامه کامپایل نخواهد شد همچنین همه کدهای باید داخل دو علامت { } نوشته شود که می تواند تو در تو باشد.

```
int i = 5;
double j = 1.23;
```

↑ ↑ ↑  
 نوع            نام            مقدار اولیه

۱-۹

برای نوشتن کد کافیسیت کدهایمان را داخل کلاس فرم بین دو آکلااد بنویسیم مثلا برای تعریف یک متغیر از نوع عدد صحیح یا متغیرهای دیگر مطابق شکل ۱-۱۰ زیر عمل می کنیم .

```
public partial class Form1 : Form
{
    //////////////////////////////////////////////////
    int i = 5;
    string j = "computer";
    double m = 4.5613;
    //////////////////////////////////////////////////
    public Form1()
    {
        InitializeComponent();
    }
}
۱۰-۱
```

متغیرهایی که در شکل بالا تعریف شده به صورت عمومی هستند برای کلاس فرم یعنی در هر جایی داخل برنامه می توانیم از آنها استفاده کنیم ولی اگر این متغیرها را داخل توابع (که در بخشهای بعدی با توابع آشنا خواهیم شد) تعریف کنیم دیگر خارج از توابع قابل استفاده نیستند.

```
.....
int i = 5;
string j = "computer";
double m = 4.5613;
//////////////////////////////////////////////////
public Form1()
{
    InitializeComponent();
}
public void fact()
{
    int n = 67;
}
```

نکته : اگر دو متغیر همنام یکی عمومی و یکی داخل تابع (محلّی) تعریف کنیم وقتی داخل تابع بخوایم از آن متغیر استفاده کنیم سی شارپ اولویت را به متغیر محلّی می دهد یعنی متغیر محلّی را استفاده میکند.

```

int i;
private void button1_Click(object sender, EventArgs e)
{
    int i;
    i
}

```

مثال: برنامه ای که دو عدد را جمع می کند و در تکست باکس قرار می دهد :

برای این کار کافیست یک دکمه و یک تکست باکس بر روی فرم برنامه قرار دهید که با فشار دکمه دو عدد را با هم جمع کند برای مقید کردن این عملیات به دکمه کافیست بر روی دکمه دوبار کلیک کنید یا از قسمت **properties** و از **Event** بر روی **Click** کلیک کنید تا رویداد کلیک دکمه فراخوانی شود یعنی وقتی این دکمه فشار داده شود این کد اجرا شود مثلا اگر خواستید وقتی موس از روی دکمه برداشته شود کدی اجرا شود باید رویداد **MouseUp** اجرا شود. وقتی رویداد کلیک را اجرا کردید کد زیر ظاهر می شود(رویداد یک تابع است)

```

private void button1_Click(object sender, EventArgs e)
{
}

```

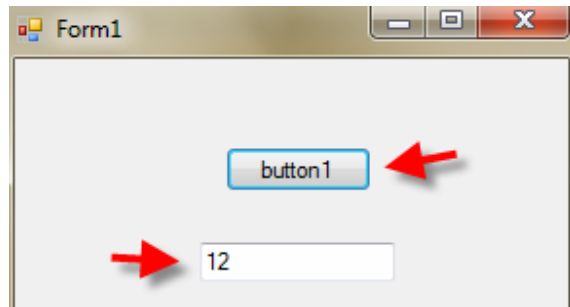
باید کد ها داخل دو **{}** نوشته شود. برای جمع کردن دو عدد کافیست دو تا متغیر تعریف کنیم و بعد مقدار دهی کنیم و حاصل جمع یا ضرب یا غیره را داخل متغیر سوم قرار دهیم و متغیر سوم را داخل تکست باکس قرار دهیم مثل کد زیر:

```

private void button1_Click(object sender, EventArgs e)
{
    int i = 5;
    int j = 7;
    int add = i + j;
    textBox1.Text = add.ToString();
}

```

حال F5 را کلیک کنید تا برنامه اجرا شود و بعد بر روی دکمه کلیک کنید تا حاصل را مشاهده کنید



نکته: برای تبدیل نوع int به string از ToString() استفاده می کنیم چون در سی شارپ متغیرهای مختلف را نمی توانیم در یکدیگر قرار دهیم مثلاً int را در string (دنباله ای از کاراکترها) و برعکس نمی توانیم قرار دهیم .

## :Char

برای تعریف کاراکتر در سی شارپ به این ترتیب عمل می کنیم

```
char i = 'a';
char j = 'z';
```

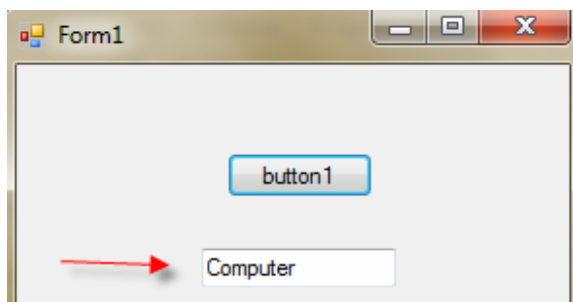
مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    char i = 'a';
    textBox1.Text = i.ToString();
}
```

## : String

اگر بخواهیم رشته ای را در سی شارپ تعریف کنیم از استرینگ استفاده می کنیم مثال:

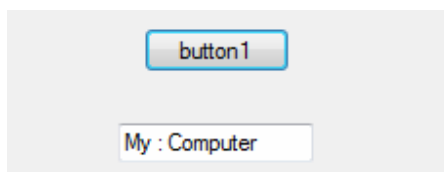
```
private void button1_Click(object sender, EventArgs e)
{
    string i = "Computer";
    textBox1.Text = i;
}
```



نکته: چون تکست باکس از نوع استرینگ است دیگر لازم نیست از `ToString()` استفاده کنیم .

نکته: جمع کردن استرینگ به معنای در کنار هم قرار دادن آنهاست نه جمع کردن مثل اعداد صحیح مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    string i = "Computer";
    string j = "My : ";
    textBox1.Text = j + i;
}
```



نکته: برای تبدیل استرینگ به `int` و دیگر متغیرها از `Convert` استفاده می کنیم : (البته استرینگ ما باید به شکل

استاندارد `int` باشد یعنی کارتهاهایمان عدد باشد مثل "۱۲۳" اگر به این صورت باشد غلط است "m۱۲۳")

```
int mynumber = Convert.ToInt32("123");
double mynumber2 = Convert.ToDouble("33.43");
Int64 mynum = Convert.ToInt64("9223372036854775807");
string str = Convert.ToString(123);
textBox1.Text = mynum.ToString();
```

نکته: تفاوت `int`, `int32`, `int64` در ظرفیت آنهاست مثلا در `int64` ما کسیم ظرفیت دو به توان شصت و سه است

و دلیل اینکه چرا نوع اعداد صحیح ما متفاوت است اینست که وقتی ما در برنامه با اعداد صحیح کوچک کار

خواهیم کرد چرا بیخودی فضای بیشتری را در حافظه اشغال کنیم .

مثال: دو عدد را از دو تکست باکس گرفته و آنها را ضرب کنید و در تکست باکس سوم قرار دهید؟ برای این کار سه تکست باکس و یک دکمه در صفحه قرار دهید و اگر خواستید text دکمه را به ضرب تغییر دهید و اگر خواستید سه تا Label قرار داده و text آنها را به عدد اول و عدد دوم تغییر دهید مثل شکل ۱۱-۱



حال بر روی دکمه ضرب کلیک کنید و کد زیر را بنویسید

```
private void button1_Click(object sender, EventArgs e)
{
    int num1 = Convert.ToInt32(textBox1.Text);
    int num2 = Convert.ToInt32(textBox2.Text);
    textBox3.Text = (num1 * num2).ToString();
}
```

برای تبدیل به این روش نیز می توانید عمل کنید

```
int num1 = int.Parse(textBox1.Text);
```

همین برنامه را برای تقسیم می نویسیم :

```
private void button1_Click(object sender, EventArgs e)
{
    int num1 = int.Parse(textBox1.Text);
    int num2 = int.Parse(textBox2.Text);
    double d3=(double)num1/num2;
    textBox3.Text = d3.ToString();
}
```



نکته : اگر خواستید اعداد اعشار شما با دو یا سه رقم اعشار نمایش داده شود باید از `string.Format` به این صورت استفاده کنید:

```
int num1 = int.Parse(textBox1.Text);
int num2 = int.Parse(textBox2.Text);
double d3=(double)num1/num2;
textBox3.Text = string.Format("{0:0.000}", d3);
```

در جدول زیر انواع نوع داده با مینیمم و ماکسیمم ظرفیت نمایش داده شده است .

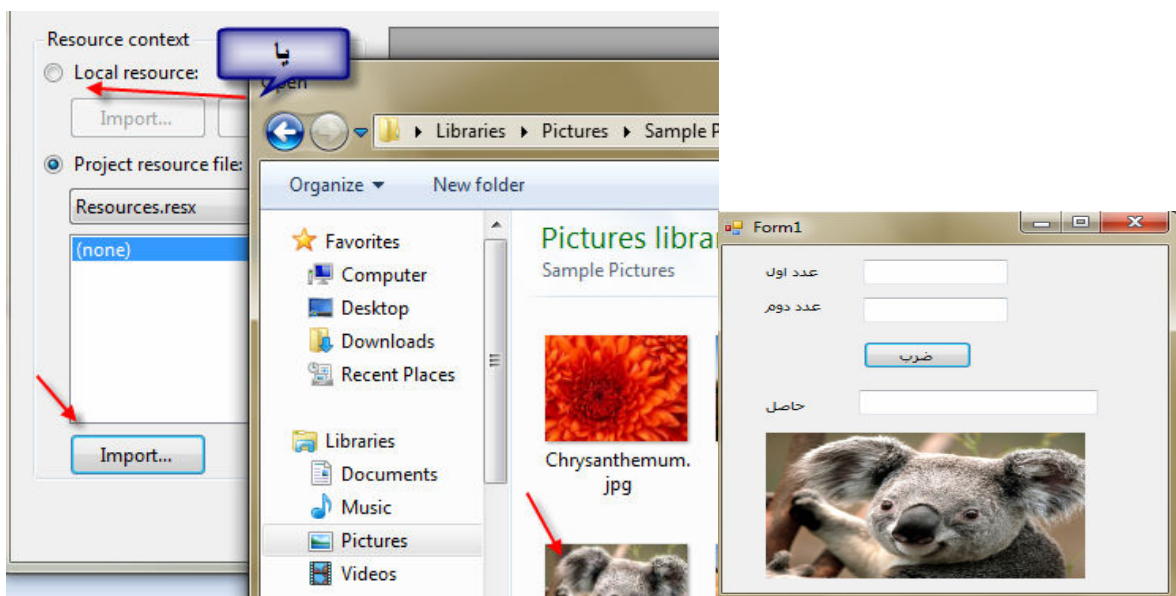
Min	Max	نوع
-۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷۵۸۰۸	۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷۵۸۰۸	Int64
-E+38۳,۴۰۲۸۲۳	E+38۳,۴۰۲۸۲۳	float
-E+308۱,۷۹۷۶۹۳۱۳۴۸۶۲۳۲	E+308۱,۷۹۷۶۹۳۱۳۴۸۶۲۳۲	double
a	z	char
true	false	bool
-۳۲۷۶۷	۳۲۷۶۷	Int16
-۲۱۴۷۴۸۳۶۴۸	۲۱۴۷۴۸۳۶۴۷	Int32,Int
.	۲۵۶	byte

## pictureBox:

برای قرار دادن یک عکس در فرم برنامه یک pictureBox در فرم قرار دهید و اندازه آن را به دلخواه تنظیم کنید حال برای قرار دادن عکس در pictureBox دو روش وجود دارد یکی اینکه از قسمت properties و از import

BackgroundImage یک عکس را

کنید



بعد BackgroundImageLayout را به Stretch تغییر دهید

و یا از طریق کد نویسی می توانید به این شکل عمل کنید و مسیر عکس را بدهید

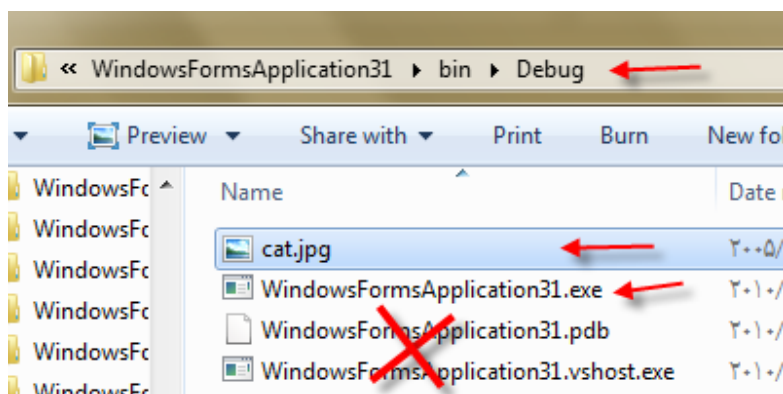
```
pictureBox1.BackgroundImage = Image.FromFile(@"c:\cat.jpg");
```

نکته: در سی شارپ از علامت \ به تنهایی نمی توانید استفاده کنید بلکه باید یا به صورت \\ و یا به صورت @" استفاده کنید.

و اگر خواستید عکس را در کنار فایل اجرایی قرار دهید باید دیگر مسیر را مشخص نکنید مانند

```
pictureBox1.BackgroundImage = Image.FromFile("cat.jpg");
```

در ضمن اگر خواستید به فایل اجرایی برنامهتون دسترسی داشته باشید و آن را در اختیار دیگران قرار دهید باید در مسیری که برنامه را ایجاد کردید از پوشه bin و Debug فایل اجرایی را بردارید که با پسوند .exe مشخص شده و چند فایل اضافی نیز وجود دارد که به درد نمی آید فقط فایلهایی که خودتان قرار دادید را هم بردارید مثل عکس



#### ۱-۴ حلقه ها :

حلقه ها از قسمت های جالب و به دردخور در برنامه نویسی و سی شارپ هستند با استفاده از حلقه ها می توان هزاران دستور را در یک خط نوشت مثلا اگر خواستید اعداد یک تا هزار را با هم جمع کنید این کار با استفاده از حلقه ها به آسانی انجام می گیرد ولی اگر حلقه ها نبود باید هزار تا `int` تعریف میکردید و یکی یکی با هم جمع می کردید. حلقه ها در سی شارپ متفاوت هستند اولین حلقه که می خواهیم به آن پردازیم حلقه `for` است نحوه تعریف حلقه به صورت شکل زیر است

```

for (int i = 0; i < 100; i++)
{
    |
}

```

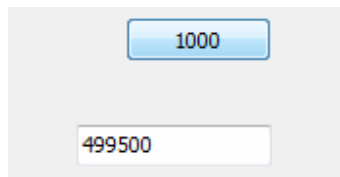
↑
↑
↑  
شروع
شرط
تمام حرکت

این حلقه صد بار اجرا می شود و در هر بار اجرا یک بار به داخل `{}` می رود و دستورات آن را اجرا می کند نحوه کارکرد حلقه به این صورت است که ابتدا مشخص می کنیم که حلقه از کجا شروع شود (مثلا اگر در بالا به جای مقدار دهی `i` با صفر از عدد ۵ استفاده میکردیم حلقه ما ۹۵ بار اجرا میشد) در اینجا مقدار اولیه `i` صفر

است حلقه میاد شرط را بررسی می کند می بیند که شرط برقرار شده است بنابراین یک واحد به  $i$  اضافه می کند و می رود داخل حلقه و دستورات را اجرا می کند بعد بر میگردد به اول حلقه حالا مقدار  $i$  به یک تغییر کرده بود باز هم شرط برقرار است یعنی یک کوچکتر از صد است بنابراین یک بار دیگر یک واحد به  $i$  اضافه می کند و وارد حلقه می شود این کار تا زمانی که شرط برقرار است ادامه میابد بنابراین حلقه ما صد بار اجرا می شود مثال: برنامه زیر اعداد یک تا هزار را با هم جمع می کند و حاصل را در تکست باکس قرار می دهد

نکته: از  $i$  می توانیم در داخل حلقه نیز استفاده کنیم

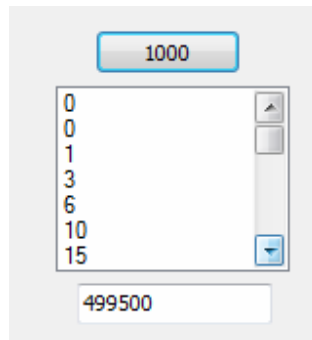
```
private void button1_Click(object sender, EventArgs e)
{
    int sum = 0;
    for (int i = 0; i < 1000; i++)
    {
        sum = sum + i;
    }
    textBox1.Text = sum.ToString();
    //1+2+3+4+5+.....+999
}
```



کته : برای توضیحات برنامه باید از علامت // استفاده کنیم وقتی این علامت را جلوی هر کد بزاریم دیگر آن کد اجرا نخواهد شد.

برای اینکه حاصل جمع را در هر مرحله ببینید و نیز با کنترل **ListBox** آشنا بشیم یک کنترل از آن بر روی فرم قرار دهید و کد بالا را به این شکل تغییر دهید:

```
int sum = 0;
for (int i = 0; i < 1000; i++)
{
    listBox1.Items.Add(sum);
    sum = sum + i;
}
textBox1.Text = sum.ToString();
```



نحوه کارکرد لیست باکس به این صورت است که هر بار یک شی به آن اضافه کردیم آن شی در یک ردیف قرار می گیرد. اگر از حلقه ها استفاده نمی کردیم با یک هزار بار این کد را می نوشتیم

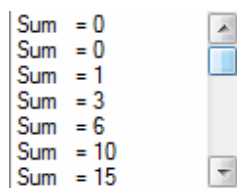
```
listBox1.Items.Add(0);
```

.....

```
listBox1.Items.Add(999);
```

حالا کد را به این شکل تغییر دهید و برنامه را اجرا کنید:

```
int sum = 0;
for (int i = 0; i < 1000; i++)
{
    listBox1.Items.Add("Sum = "+sum);
    sum = sum + i;
}
textBox1.Text = sum.ToString();
```



حال یک comboBox در صفحه قرار دهید و کد زیر را بنویسید تا تفاوت آن را با لیست باکس بفمید:

```
int sum = 0;
for (int i = 0; i < 1000; i++)
{
    listBox1.Items.Add("Sum = "+sum);
    comboBox1.Items.Add("Sum = " + sum);
    sum = sum + i;
}
textBox1.Text = sum.ToString();
```

مثال: می خواهیم یک برنامه بنویسیم که در یک لیست باکس جستجویی انجام بدهیم ؟

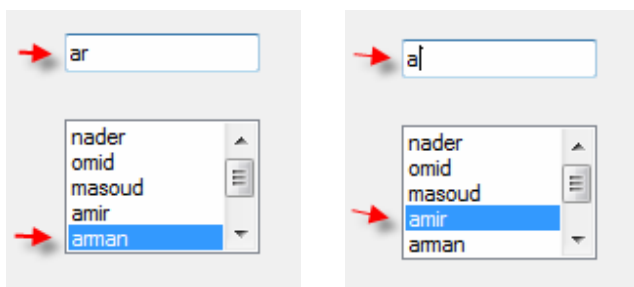
برای اینکار یک تکست باکس و یک لیست باکس در صفحه قرار دهید. بر روی فرم کلیک کنید تا رویداد Load فرم فراخوانی شود این رویداد موقعی کار می کند که فرم می خواهد اجرا شود و بالا بیاد بنابراین وقتی فرم می خواهد لود شود ما لیست باکس را با رشته هایی پر می کنیم مثل کد زیر:

```
private void Form1_Load(object sender, EventArgs e)
{
    listBox1.Items.Add("nader");
    listBox1.Items.Add("omid");
    listBox1.Items.Add("masoud");
    listBox1.Items.Add("amir");
    listBox1.Items.Add("arman")
}
}
```

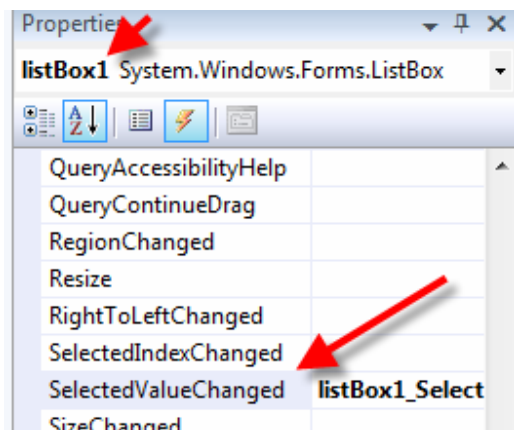
حال بر روی تکست باکس کلیک کنید تا رویداد TextChanged فراخوانی شود همانطور که قبلا مشاهده فرمودید رویداد کلیک دکمه را فراخوانی کردیم و آن رویداد زمانی اجرا می شد که بر روی دکمه کلیک کنیم ولی رویداد TextChanged زمانی اجرا می شود که مقداری را در تکست باکس وارد کنیم و یا آن را تغییر دهیم. خوب حالا کد زیر را در درون رویداد TextChanged بنویسید:

```
private void textBox1_TextChanged(object sender, EventArgs e)
{
    listBox1.SelectedIndex = listBox1.FindString(textBox1.Text);
}
}
```

حال برنامه را اجرا کنید بعد در تکست باکس یک کاراکتر وارد کنید مثلا اگر کاراکتر a را وارد کنید نامی را پر رنگ می کند که نام اول آن a است اگر دومین کاراکتر را هم وارد کردید ممکن در دومین کاراکتر انتخاب لیست باکس عوض شود



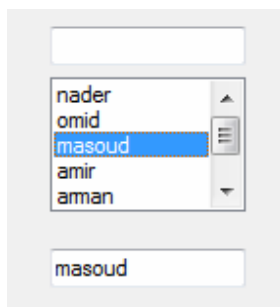
اگر خواستید عنصری را که در لیست باکس انتخاب می کنید برای شما برگشت داده شود و از آن استفاده کنید باید از رویداد SelectedValueChanged استفاده کنید. برای فراخوانی این رویداد بر روی آن از قسمت رویداد لیست باکس کلیک کنید تا این رویداد برای شما ایجاد شود مانند شکل زیر:



در کد زیر عنصری انتخابی در یک تکست باکس قرار می گیرد:

```
private void listBox1_SelectedValueChanged(object sender, EventArgs e)
{
    textBox2.Text = listBox1.SelectedItem.ToString();
}
```

حال برنامه را اجرا کنید و بر روی یکی از نامها کلیک کنید تا در تکست باکس دوم نمایش داده شود.



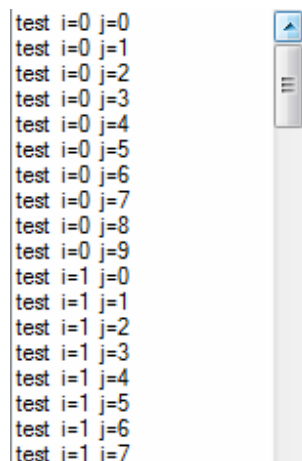
حلقه های تو در تو: در این حلقه ها به ازای اجرای یک بار در حلقه خارجی حلقه داخلی به طور کامل اجرا

میشود

: مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            listBox1.Items.Add("test i="+i+" j="+j);
        }
    }
}
```

فرض کنید حلقه داخلی وجود ندارد روش کار به این صورت است که حلقه ده بار دستورات داخلی خود را اجرا می کند حال اگر آن دستور داخلی یک حلقه باشد موظف است که به طور کامل اجرا شود بنابراین در هریک بار مراجعه به داخل حلقه ، حلقه داخلی ده بار اجرا میشود در کل این حلقه  $10 \times 10$  بار اجرا می شود بنابراین لیست باکس صد بار کلمه تست را در خود نمایش می دهد.



```
test i=0 j=0
test i=0 j=1
test i=0 j=2
test i=0 j=3
test i=0 j=4
test i=0 j=5
test i=0 j=6
test i=0 j=7
test i=0 j=8
test i=0 j=9
test i=1 j=0
test i=1 j=1
test i=1 j=2
test i=1 j=3
test i=1 j=4
test i=1 j=5
test i=1 j=6
test i=1 j=7
```

توجه : سعی کنید در برنامه هایتان کمتر از حلقه تو در تو استفاده کنید چون دارای زمان اجرای طولانی هستند. در ضمن حلقه ها را به صورت زیر نیز می توانید تعریف کنید که معادل هم هستند: (تعریف `int` خارج از حلقه)

```
int i, j;
for ( i = 0; i < 10; i++)
{
    for ( j = 0; j < 10; j++)
    {
        listBox1.Items.Add("test i="+i+" j="+j);
    }
}
```



واگر خواستید گام حرکتان را تغییر دهید مثلا گام حرکت دوبرابر شود می توانید به صورت  $i+=2$  عمل کنید. همانطور که می دانید  $i++$  معادل است با  $i+=1$  در جدول زیر معادل تعدادی از کدهای با هم آورده شده است :

$I+=1$	$I++$	$I=I+1$
$I-=1$	$I--$	$I=I-1$
$I+=N$		$I=I+N$
$I*=N$		$I=I*N$
$I/=N$		$I=I/N$
$I-=N$		$I=I-N$
N معادل هر عدد می تواند باشد		

حلقه ها را می توانید برعکس هم بنویسید مثل:

```
for ( i = 10; i > 0; i--)
```

دومین حلقه ای که می خواهیم با آن آشنا بشیم حلقه While است نحوه تعریف آن به این صورت است

```
while (true)
{
}
```

تازمانی که عبارت داخل پرانتز درست (true) باشد حلقه ادامه می یابد در اینجا گام حرکت در داخل {} باید شارژ شود اگر این کار را نکنید حلقه بی نهایت بار اجرا می شود که در بعضی برنامه ها هم همین روش استفاده می شود. برای فهمیدن بهتر این برنامه مثال قبلی را با همین روش حل می کنیم :

```
int i = 0;
while (i<100)
{
    listBox1.Items.Add("test");
    i++;
}
```

نکته : در هر حلقه اگر خواستید در جایی از برنامه از حلقه خارج شوید از دستور break استفاده می شود  
مثلا در همین مثال اگر خواستید وقتی i برابر ۵۰ شد از حلقه بیرون بیاید به همین روش عمل می کنیم مثال:

```
int i = 0;
while (i<100)
{
    listBox1.Items.Add("test");
    i++;
    if (i == 50)
    {
        break;
    }
}
```

البته هنوز به شرط ها نرسیدیم که در قسمت بعدی آنها را توضیح خواهیم داد.

حلقه دیگری که می خواهیم به آن پردازیم حلقه do while است روش کار این حلقه مثل حلقه while است با این تفاوت که در هر مرحله اول کد داخل do اجرا می شود و بعد شرط بررسی می شود.

```
do
{
} while (true);
```

مثال :

```
private void button1_Click(object sender, EventArgs e)
{
    int i = 0;
    do
    {
        listBox1.Items.Add(i);
        i++;
    } while (i < 3);
}
```

## ۱-۵ شرط ها IF :

اگر خواستید در برنامه شرطی را اعمال کنید از دستور if استفاده می کنیم مثلا ادامه کد برنامه ای را می توانید با شرط کنترل کنید. برای شرط تساوی از علامت دو مساوی == استفاده می شود و برای شرط بزرگتر یا

کوچکتر از دو علامت <> و برای بزرگتر مساوی و یا کوچکتر مساوی از دو علامت <= و >= استفاده می شود و برای نقیض از علامت != استفاده می شود.

مثال: برنامه بنویسید که عدد را از تکست باکس اول بخواند و تشخیص دهد که آیا عدد مثبت است یا منفی؟

```
private void button1_Click(object sender, EventArgs e)
{
    int i = Convert.ToInt32(textBox1.Text);
    if (i < 0)
    {
        textBox2.Text = "منفی";
    }
    if (i > 0)
    {
        textBox2.Text = "مثبت";
    }
    if (i == 0)
    {
        textBox2.Text = "عدد صفر است";
    }
}
```



در این روش یکی یکی شرط ها بررسی می شود و هر کدام درست بود آن را انجام می دهد ولی اینکار سرعت برنامه را می آورد پایین بنابراین ما از روش دیگر (else if) استفاده می کنیم در این روش هر کجا شرط برقرار شد دیگر به سراغ شرط بعدی نمی رود. مثال بالا را با همین روش حل می کنیم

```
private void button1_Click(object sender, EventArgs e)
{
    int i = Convert.ToInt32(textBox1.Text);
    if (i < 0)
    {
        textBox2.Text = "منفی";
    }
    else if (i > 0)
    {
        textBox2.Text = "مثبت";
    }
    else if (i == 0)
    {
        textBox2.Text = "عدد صفر است";
    }
}
```

```

    {
        textBox2.Text = " است صفر عدد ";
    }
}

```

معادل `else if` دستور سویچ (`switch`) است که هر کجا شرط برقرار شد کد آن بلوک را اجرا می کند و با دستور `break` خارج می شود

مثال: برنامه ای که اعداد را دریافت می کند و معادل فارسی آنها را در تکست باکس دوم قرار می دهد؟

```

private void button1_Click(object sender, EventArgs e)
{
    int i = Convert.ToInt32(textBox1.Text);
    switch (i)
    {
        case 0:
            textBox2.Text = "صفر";
            break;
        case 1:
            textBox2.Text = "یک";
            break;
        case 3:
            textBox2.Text = "سه";
            break;
        default:
            textBox2.Text = "کدام هیچ";
            break;
    }
}

```

اگر خواستید می توانید از `default` در دستور سویچ استفاده کنید که در صورت برقرار نبودن هیچ کدام از شرط ها دستور داخل `default` اجرا می شود.

نکته : عبارت داخل پرانتز هر نوعی باشد باید نوع مقابل `case` هم از همان نوع باشد مثال اگر داخل پرانتز از نوع استرینگ باشد عبارت داخل پرانتز هم باید از نوع استرینگ باشد. مثال بالا را بدون تبدیل به عدد صحیح تغییر می دهیم:

```

private void button1_Click(object sender, EventArgs e)
{
    // int i = Convert.ToInt32(textBox1.Text);
    string str = textBox1.Text;
    switch (str)
    {
        case "0":
            textBox2.Text = "صفر";
    }
}

```

```

        break;
    case "1":
        textBox2.Text = "یک";
        break;
    case "3":
        textBox2.Text = "سه";
        break;
    case "hello":
        textBox2.Text = "سلام";
        break;
    default:
        textBox2.Text = "هیچ کدام";
        break;
    }
}

```

البته داخل پرانتز `switch` را به طریق زیر نیز می توانیم بنویسیم:

```
switch (textBox1.Text)
```

اگر خواستید چند شرط را همزمان چک کنید مثلا یا این اتفاق بیافتد یا آن ، این عمل را انجام بدهد. برای اینکار از علامات `||` برای OR (یا) و از `&&` برای AND (و) استفاده می شود

مثال: اگر اعداد ما بین صفر و ده باشد بگوید تک رقمی است و اگر اعداد ما صد یا دویست باشد بگوید اعداد صدتایی است ؟

```

private void button1_Click(object sender, EventArgs e)
{
    int i = Convert.ToInt32(textBox1.Text);
    if (i > 0 && i < 10)
    {
        textBox2.Text = "رقمی تک";
    }
    else if (i == 100 || i == 200)
    {
        textBox2.Text = "صدتایی";
    }
    else
    {
        textBox2.Text = " ";
    }
}

```



اگر بگوییم اعداد ما مخالف یک چیز باشد مثلاً مخالف عدد ۵۰۰ از این روش استفاده می‌کنیم

```
If (i != 500) {
}
```

در زیر به چندی مثال می‌پردازیم تا بهتر با این دستورات آشنایی یابید:

۱- برنامه‌ای که ماکسیمم سه عدد را تشخیص می‌دهد؟

```
int a, b, c, max;
a = int.Parse(textBox1.Text);
b = int.Parse(textBox2.Text);
c = int.Parse(textBox3.Text);
max = a;
if (b > max)
{
    max = b;
}
if (c > max)
{
    max = c;
}
textBox4.Text = max;
```

۲- یک عدد را از ورودی گرفته و کامل بودن آن را تشخیص دهد:

```
private void button1_Click(object sender, EventArgs e)
{
    int x, i, sum = 0;
    x = int.Parse(textBox1.Text);
    for (i = 2; i <= x; i++)
    {
        if (x % i == 0)
        {
            sum = sum + (x / i);
        }
    }
    if (sum == x)
        textBox2.Text = "است کامل عدد";
    else
```

```
textBox2.Text = "نیست کامل عدد";
```

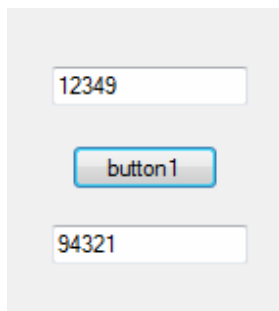
```
}
```

نکته : از علامت / برای بدست آوردن خارج قسمت و از علامت % برای بدست آوردن باقی مانده استفاده می شود.

۳- یک عدد را از ورودی گرفته و برعکس آن را نمایش دهد:

```
private void button1_Click(object sender, EventArgs e)
{
    int num, x = 0;
    string str = "";
    num = int.Parse(textBox1.Text);
    while (num != 0)
    {
        x = num % 10;
        num = num / 10;
        str += x;
    }

    textBox2.Text = str;
}
```



۴- بین صفر تا هزار اعدادی را که بر دو بخش پذیر نیستند را در یک لیست باکس قرار دهد :

```
private void button1_Click(object sender, EventArgs e)
{
    for (int i = 0; i < 1000; i++)
    {
        if (i % 2 != 0)
            listBox1.Items.Add(i);
    }
}
```

۵- برنامه ای که سه عدد را خوانده اگر آنها بتوانند اعضای یک مثلث باشند تشخیص دهد:

```
private void button1_Click(object sender, EventArgs e)
{
    float a, b, c;
    a = float.Parse(textBox1.Text);
```

```

b = float.Parse(textBox2.Text);
c = float.Parse(textBox3.Text);
if (a + b > c && a + c > b && b + c > a)
    textBox4.Text = "OK";
else
    textBox4.Text = "NO";
}

```

نکته: از float هم می توانید برای اعداد اعشاری استفاده کنید.

۶- برنامه ای که ریشه های معادله درجه دوم را بدست می آورد:

```

private void button1_Click(object sender, EventArgs e)
{
    double x1, x2, delta, A, B, C;
    A = double.Parse(textBox1.Text);
    B = double.Parse(textBox2.Text);
    C = double.Parse(textBox3.Text);
    delta = B * B - (4 * A * C);
    if (delta > 0)
    {
        x1 = (-B + Math.Sqrt(delta)) / 2 * A;
        x2 = (-B - Math.Sqrt(delta)) / 2 * A;
        textBox4.Text = ("x1= " + x1 + " x2= " + x2);
    }
    if (delta < 0)
        textBox4.Text = "ندارد ریشه";
}

```

۷- برنامه فاکتوریل:

```

int i, n, factoriel = 1;
n = int.Parse(textBox1.Text);
for (i = 1; i <= n; i++)
{
    factoriel = factoriel * i;
}
textBox2.Text = factoriel.ToString();

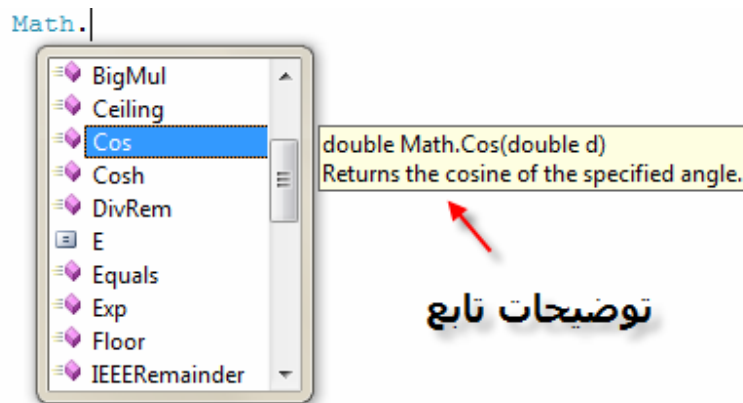
```

: Math 1-6

در این کلاس بعضی از توابع ریاضی وجود دارد که می توانید از آنها استفاده کنید مثلا می توانید سینوس و

کوسینوس و جذر و توان یک عدد را بدست آورید:





### ۷-۱ توابع در سی شارپ :

هدف از ایجاد تابع در برنامه نویسی انتقال کدها به داخل آن با نام مشخص و فراخوانی آن توسط برنامه و دیگر توابع در صورت نیاز است. مثلاً می‌توانیم تابعی بنویسیم که فاکتوریل یک عدد را برایمان حساب کند و هر وقت آن تابع را فراخوانی کردیم این عمل را برایمان انجام می‌دهد. تابع دارای مزایای بسیاری است مثلاً فرض کنید که دو دکمه در صفحه وجود دارند که هر کدام کار بخصوصی انجام می‌دهند ولی هر کدام در جایی از کد نویسیشان نیاز به حساب فاکتوریل دارند بنابراین هر کدام می‌توانند به طور جداگانه تابع فاکتوریل را فراخوانی کنند بدون آنکه نیاز باشد هر کدام باز هم دوباره شروع به کد نویسی فاکتوریل کنند فقط یک بار تابع می‌نویسیم که عمل فاکتوریل را انجام بدهد هر چند بار خواستیم می‌توانیم از آن استفاده کنیم.

تابع‌ها انواع مختلف دارند مثلاً در همین تابع فاکتوریل نیاز است که اعداد را بفرستیم به تابع و آن کار را انجام بدهد و مقداری را برگرداند و بعضی توابع نیز هستند که هیچ مقداری نمی‌گیرند و هیچ مقداری را نیز برگشت نمی‌دهند و ممکن است هیچ مقداری نگیرد ولی یک مقدار برگشت بدهد و نیز ممکن است یک مقدار بگیرد و هیچ مقداری برگشت ندهد در زیر به توضیح انواع تابعها با مثال می‌پردازیم.

نکته: تابعها باید خارج از دیگر توابع تعریف شوند و در دیگر توابع فراخوانی شوند. توابع می‌توانند خودشان را نیز فراخوانی کنند.

تابعی که هیچ مقداری را نمی‌گیرد و هیچ مقداری برگشت نمی‌دهد به صورت شکل ۱-۱۲ تعریف می‌شود

```
void fact ()
{
}
نام تابع
```

۱۲-۱

تابعی که مقداری از انواع مختلف و به تعداد مختلف می گیرد ولی هیچ مقداری برگشت نمی دهد

```
void fact(int i, int j)
{
    //code
}
void fact2(int i, string st, double b)
{
    //code
}
```

تابعی که مقداری از انواع مختلف می گیرد و از انواع مختلف بر می گرداند

```
int fact(int i, int j)
{
    //code
    return
}
string fact2(int i, string st, double b)
{
    //code
    return
}
```

مثال : برنامه ای بنویسید که با استفاده از دو دکمه دو مقدار متفاوت فاکتوریل تولید کند ؟

در این مثال هر دو دکمه ، یک تابع را با مقادیر متفاوت فراخوانی می کنند و مقدار برگشتی را در تکست باکس قرار می دهند

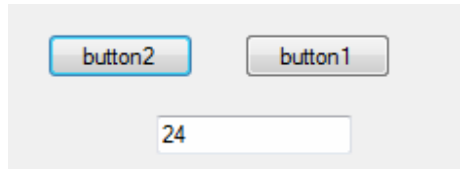
```
private void button1_Click(object sender, EventArgs e)
{
    textBox2.Text = fact(5).ToString();
}

private void button2_Click(object sender, EventArgs e)
{
    textBox2.Text = fact(4).ToString();
}
////////////////////////////////////
////////////////////////////////////
int fact(int num)
{
    int factoriel = 1, i;
```

```

for (i = 1; i <= num; i++)
{
    factoriel = factoriel * i;
}
return factoriel;
}
//£////////////////////////////////////
////////////////////////////////////

```



اگر از تابع استفاده نمی کردیم باید هر دو دکمه به طور جداگانه کد مربوط به فاکتوریل را به طور کامل می نوشتند.

حال همین برنامه را برای تابع بدون برگشتی می نویسیم :

```

private void button1_Click(object sender, EventArgs e)
{
    fact(5);
}

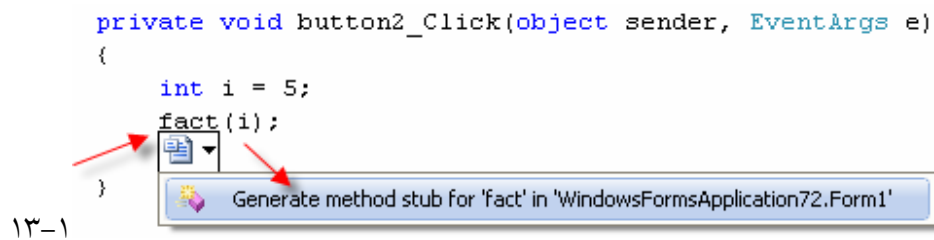
private void button2_Click(object sender, EventArgs e)
{
    fact(4);
}

////////////////////////////////////
////////////////////////////////////
void fact(int num)
{
    int factoriel = 1, i;
    for (i = 1; i <= num; i++)
    {
        factoriel = factoriel * i;
    }
    textBox2.Text = factoriel.ToString();
}

```

نکته : توابع فقط یک مقدار بر می گردانند ولی بعدا یاد خواهیم گرفت که چگونه چند مقدار را هم برگشت بدیم .

روش دیگر برای ایجاد تابع: کفیسست نام تابع مورد نظر همراه با پارامترهای ارسالی آن را در صورت نیاز بنویسیم بعد علامتی زیر نام تابع ما ظاهر خواهد شد با کلیک بروی آن تابع ما به طور اتوماتیک ایجاد خواهد شد مثل شکل ۱۳-۱



```
private void button2_Click(object sender, EventArgs e)
{
    int i = 5;
    fact(i);
}

private void fact(int i)
{
}

```

نکته: نیازی به نوشتن `private` و یا `public` جلوی تابع نیست ولی در بحث کلاس ها نیاز است که به آنها می پردازیم فقط به این نکته اشاره می کنم که اگر تابع و یا متغیر هایمان را از نوع `private` تعریف کنیم دیگر کلاس ها نمی توانند از این تابع ما استفاده کنند.

حال اگر تابع ما مانند شکل ۱۴-۱ فراخوانی می شد سی شارپ می فهمید که تابع به صورت بازگشتی است و از چه نوعی و تابع را به صورت بازگشتی ایجاد می نمود در این صورت باید مقداری برگشت می دادیم.

```

private void button2_Click(object sender, EventArgs e)
{
    int i = 5;
    → int j = fact(i);
    → string str2 = fact2(i);
}

private string fact2(int i)
{
    return "120";
}

private int fact(int i)
{
    return 120;
}
۱۴-۱

```

همانطور که ملاحظه کردید توابع به سادگی ایجاد ومورد استفاده قرار میگیرند.

### توابع بازگشتی:

در توابع بازگشتی توابع خودشان را فراخوانی می کنند تا شرط داخل توابع بر قرار باشد. مثال بالا را با استفاده از تابع بازگشتی انجام می دهیم بعد آن را توضیح می دهیم:

```

private void button2_Click(object sender, EventArgs e)
{
    textBox2.Text = fact(4).ToString();
}
////////////////////////////////////
////////////////////////////////////
int fact(int num)
{
    if (num == 1)
        return num;
    else return (num * fact(num - 1));
}

```

فرض کنید عدد چهار به تابع فرستاده شده است به خط `if` می رسد چون چهار مساوی یک نیست به `else` مراجعه می کند و تابع خودش را فراخوانی می کند و در عدد ۴ ضرب می کند

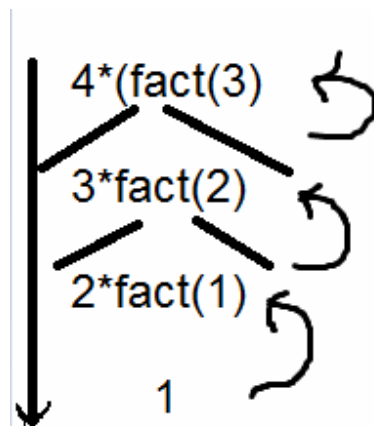
$4 * \text{fact}(4-1)$

باز وارد تابع می شود و شرط برقر نیست به else می رود و کد زیر اجرا می شود

$$3 * \text{fact}(2)$$

و در مرحله بعد کد زیر اجرا می شود  $2 * \text{fact}(1)$

حال شرط برقرار است تابع مقدار یک را بر می گرداند و در ۲ ضرب می شود و حاصل در ۳ و ۴ ضرب شده و مقدار ۲۴ برگشت داده می شود دیاگرام این برنامه می تواند به صورت زیر باشد.



$$4 * 3 * 2 * 1$$

می توان به زبان ریاضی نیز این را اثبات کرد با فرض  $\text{num}=n$  و  $\text{fact}=T$

$$\begin{aligned} & n * T(n-1) \\ & n * (n * T(n-1-1)) \\ & n * (n * (n * T(n-1-1))) = \\ & (n^3) * T(n-3) = n! \end{aligned}$$

۸-۱ آرایه ها :

حال این سوال پیش می آید اگر بخواهیم تعداد زیادی از نوع های مختلف درکنار یکدیگر داشته باشیم چکار باید کنیم مثلا اگر خواستیم هزار نام را نگه داریم باید هزار تا استرینگ تعریف کنیم که این اصلا خوب نیست

بنابراین باید سراغ آرایه برویم . برای ساختن آرایه ها کفایست اول نوع آرایه را مشخص کنید مثلا از نوع صحیح است یا کاراکتر و یا رشته و یا هر نوع دیگر. بعد دوتا علامت براکت می زاریم بعد نام آرایه را مشخص میکنیم بعد طول آرایه را مشخص می کنیم. در شکل ۱-۱۵ نحوه ایجاد انواع آرایه و مقدار دهی آنها را مشاهده می کنید برای مقدار دهی کفایست اندیس آرایه را مشخص کنید و مقدار مورد نظر را در آن قرار دهید. فقط مواظب باشید اندیس مورد نظر خارج از بازه ای که تعریف کردید نباشد.

```

////////// ایجاد انواع آرایه یگه بعدی
int[] myarray1 = new int[10];
double[] myarray2 = new double[55];
char[] myarray3 = new char[101];
string[] myarray4 = new string[1000];
long[] myarray5 = new long[8]; //or
Int64[] myarray6 = new Int64[4];

////////// مقدار دهی
myarray1[0] = 5232;
myarray2[33] = 2.3333;
myarray3[94] = 'a';
myarray4[555] = "cat"; //or
myarray4[556] = "گربه";
myarray5[1] = 9223372036854775807; //max
myarray6[3] = 9223372036854775807; //max

```

۱۵-۱ //

حال اگر آرایه ها و حلقه ها در کنار یکدیگر قرار گیرند می توانیم با یک حلقه و یک آرایه هم آرایه را مقدار دهی کنید و هم آن را بخوانید. موقع استفاده از آرایه ها نیز کفایست با استفاده از اندیس به مقادیر آن دست یافت مثلا در مثال زیر عنصر پنجم آرایه را در یک تکس باکس قرار میدهیم.

Textbox1.text=myarray4[5];

مثال: آرایه ای از استرینگ ها به طول صد تعریف می کنیم و بعد مقدار دهی و سپس آن را در یک لیست باکس قرار میدهیم.

```

private void button1_Click(object sender, EventArgs e)
{
    string[] str = new string[100];

```

```

for (int j = 0; j < 100; j++)
{
    str[j] = "Computer = " + j;
}
for (int j = 0; j < 100; j++)
{
    listBox1.Items.Add(str[j]);
}
}

```

وقتی شما آرایه ای تعریف می کنید در واقع فضای Ram کامپیوتر خود را به اندازه آرایه ای که تعریف می کنید اشغال می کنید مثال اگر فضای رم شما قبل اجرای کد زیر به شکل ۱-۱۶مقابل باشد



1-16

```

private void button1_Click(object sender, EventArgs e)
{
    int[] str = new int[1000000000];
    for (int i = 0; i < str.Length; i++)
        str[i] = 4444444;
}

```

بعد از اجرای کد زیر فضای رم نزدیک ۳۰۰ مگابایت افزایش می یابد.



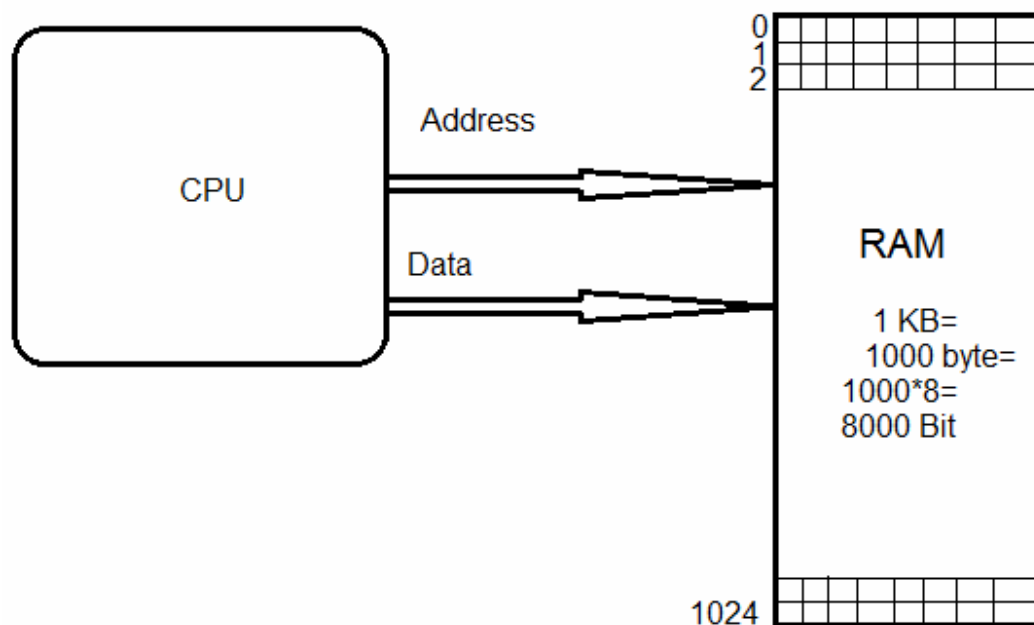
در اینجا می خواهیم شکل ساده ای از رم بیارم و توضیحاتی درباره ساختار داخلی آن و ارتباط آن با cpu بدم. همانطور که در شکل ۱-۱۷ مشاهده می کنید cpu از طریق دو درگاه آدرس و دیتا به رم وصل شده است رمی که در شکل کشیده شده است دارای ظرفیت یک کیلو بایت است (خیلی کم) وقتی ما می خواهیم دیتایی مثل int را در رم ذخیره کنیم اطلاعات را (cpu) بر روی گذرگاه دیتا قرار میدهد و یک آدرس نیز بر روی گذرگاه آدرس قرار می دهد مثلا خانه ۵۰۰ و در آن ذخیره می کند (البته تشخیص خانه ها خالی و اینکه در کجای رم



ذخیره شود از وظایف سیستم عامل است ) و هنگامی که ما آرایه ای تعریف می کنیم مثل آرایه ای به طول ۱۰۰ یعنی در رم صد خانه یا (ممکن است بیشتر باش) اشغال می کنیم. البته وقتی آرایه را مقدار دهی کردیم فضا در رم اشغال می شود.

نکته: در رم های جدید این فضا ها بسیار افزایش پیدا کرده اند .

در این شکل هر خانه ۸ بیت است و ما وقتی یک نوع از int32 تعریف می کنیم در واقع در این نوع رم ۴ ردیف اشغال می کنیم (8\*4=32bit) !!!!



۱۷-۱

درسی شارپ حلقه ها مانند زبان سی پیاده سازی شده اند اما حلقه جالبی در سی شارپ وجود دارد که از آن برای گردش در میان آرایه ها استفاده میشود نام این حلقه

## Foreach

است نحوه عملکرد این حلقه به این صورت است که بر روی تک تک عناصر آرایه تا پایان آن حرکت می کند [2] و در هر بار حرکت بر روی این عناصر یک بار هم به داخل آکولاد خود می رود و می تواند کدی که در داخل آن نوشته شده را اجرا کند و حتی عنصری که در هنگام رفتن به داخل بر روی آن قرار داشت در داخل

خود استفاده کند. در شکل ۱-۱۸ نحوه استفاده از آن نمایش داده شده است. از جمله مزایای این حلقه این است که لازم نیست طول آرایه را بدانیم و نیز برای آرایه های دوبعدی و چند بعدی لازم نیست حلقه ها تودر تو بنویسیم

```

////////////////////////////////////
foreach (string str1 in myarray4)
{
    MessageBox.Show(str1);
}
//or
foreach (int num in myarray1)
{
    string st=num.ToString();
    MessageBox.Show(st);
}

```

۱-۱۸

معادل کد بالا با حلقه for ....

```

for (int i = 0; i < 1000; i++)
{
    string str1 = myarray4[i];
    MessageBox.Show(str1);
}

```

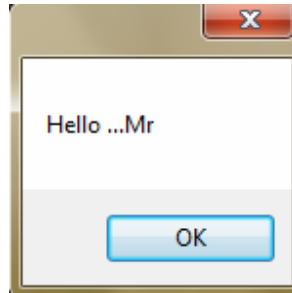
نکته : اگر خواستید پیغامی را به کاربر در قالب کادر نمایش داده شود از `MessageBox.Show` استفاده می شود همچنین اگر خواستید سوالی از کاربر پرسیده شود و با توجه به درخواست کاربر کاری انجام شود مانند پاک کردن فایل که در ویندوز مشاهده می کنید از `MessageBox.Show` استفاده می شود.

مثال:

```

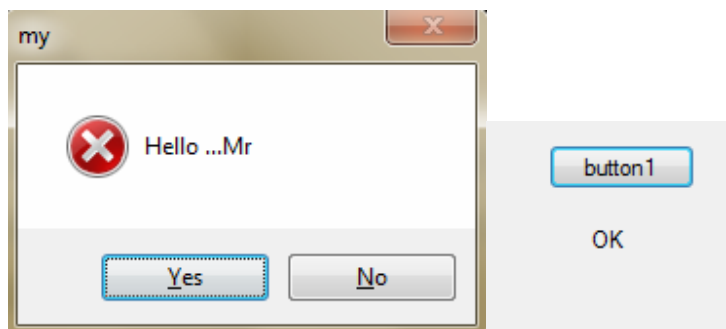
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello ...Mr");
}

```

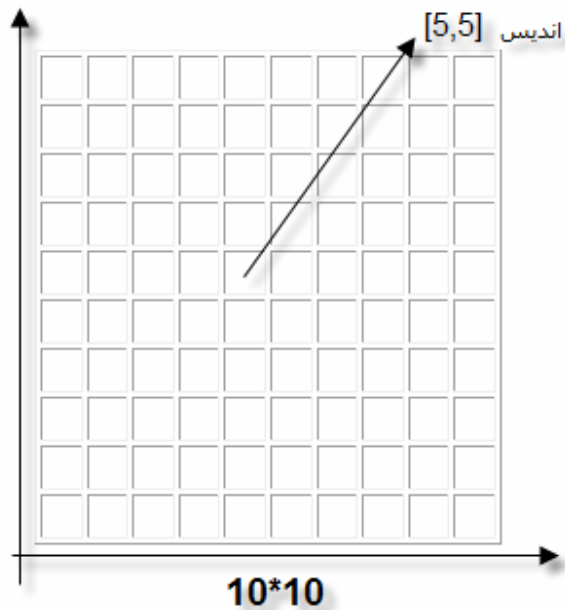


یا

```
private void button1_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Hello ...Mr", "my", MessageBoxButtons.YesNo,
        MessageBoxIcon.Error)== DialogResult.Yes)
    {
        label1.Text = "OK";
    }
    else
        label1.Text = "NO";
}
```



آرایه چند بعدی: آرایه چند بعدی مانند یک ماتریس چند بعدی عمل میکند مثلا آرایه دو بعدی مانند یک ماتریس دو بعدی است که در راستای محور X ها و Y ها امتداد یافته.



البته چون درکدنویسی اندیس آرایه از صفر شروع می شود در دسترسی به خانه [5,5] باید یک اندیس کمتر ذکر کنیم مثل [4,4]

برای تعریف آرایه دو بعدی و مقدار دهی و خواندن آن مانند شکل زیر عمل می کنیم:

```
private void button1_Click(object sender, EventArgs e)
{
    int[,] ms = new int[10, 12];
    string[,] str1 = new string[100, 108];
    str1[1, 4] = "Csharp";
    ms[9, 9] = 123;
    textBox1.Text = ms[9, 9].ToString();
    textBox2.Text = str1[1, 4];
}
```

نکته: برای مقدار دهی آرایه چند بعدی و خواندن آنها می توانید از حلقه های تو در تو استفاده کنید مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    int[, ,] myint = new int[10, 10, 10];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            for (int k = 0; k < 10; k++)
            {
                myint[i, j, k] = i + j + (2 * k);
            }
        }
    }
}
```

```

    }
}

```

اگر نحوه خواندن برای شما مهم نیست می توانید از حلقه `foreach` استفاده کنید:

```

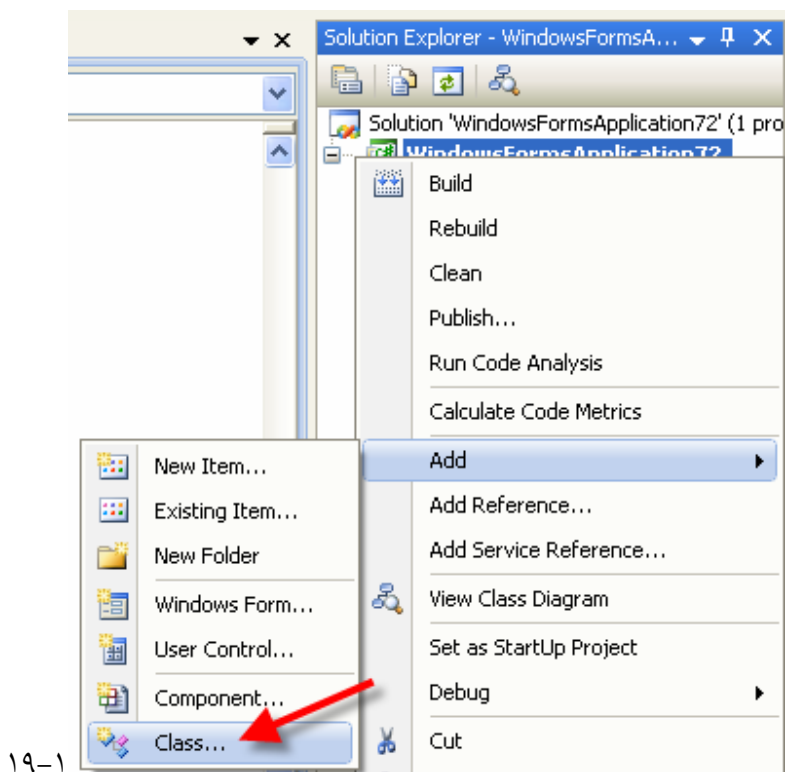
foreach (int i in myint)
{
    listBox1.Items.Add(i);
}

```

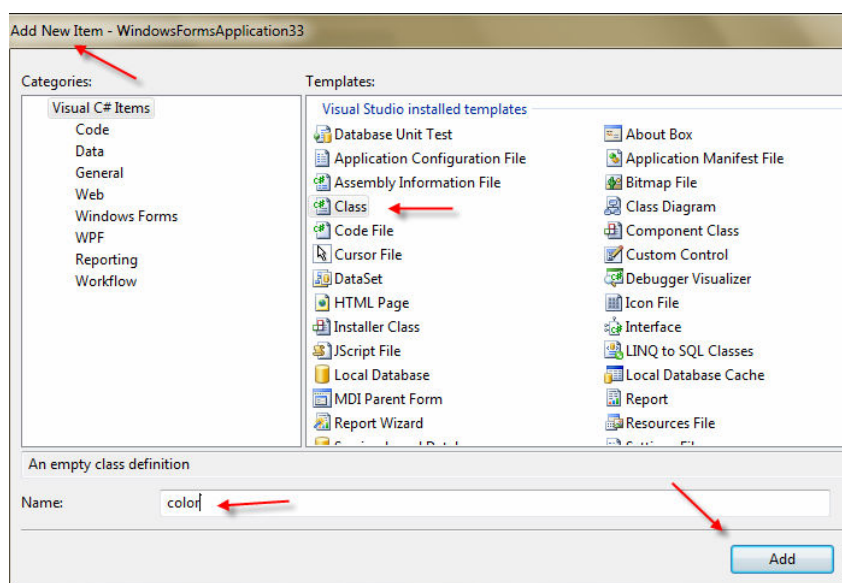
## ۹-۱ کلاس ها در سی شارپ :

برای فهم کلاس ها بهتر است مثال کارخانه ماشین را برایتان بزنم. فرض در یک کارخانه ماشین اتاق هایی وجود دارند که در هر کدام قسمتی (قطعه ای) از ماشین به تعداد زیادی وجود دارد مثلا اتاقی وجود دارد که در آن تعداد زیادی لاستیک و در اتاق دیگر تعداد زیادی موتور وجود دارد حالا اینکه چگونه اینها ساخته شده اند کاری به آنها نداریم خوب حالا فرض کنید کارخانه می خواهد صد تا ماشین به بیرون صادر کند . برای اینکار می تواند به راحتی با قرار دادن صد تا از این قطعات در کنار هم صد تا ماشین به راحتی تولید کند و وقتی هم که قطعه ای از ماشین مانند موتور ایراد در آورد به سادگی می تواند آن را در آورده و تعمیر کند این نیست که برای تعمیر موتور اول تایرها را در بیارد بعد بدنه ماشین را بعد به موتور برسد در زبان برنامه نویسی هم ما به این صورت عمل می کنیم یعنی یک کلاس تعریف می کنیم و کدهایمان را درون آن کلاس می نویسیم مثلا کلاس بدنه و یا رنگ ماشین که هر وقت به آن نیاز داشتیم یکی از آنها را بر می داریم و استفاده می کنیم به این برداشتن قطعه در برنامه نویسی شی گرای (object-oriented) می گویند یعنی وقتی ما خواستیم از قطعه ای استفاده کنیم یک شی از آن تعریف می کنیم و با استفاده از آن شی به خصوصیات داخلی آن کلاس دسترسی پیدا می کنیم . این تعریف ساده ای بود که برای کلاس گفتم بهتر است یک برنامه جدید ایجاد کنیم و از کلاس در آن برنامه استفاده کنیم در این برنامه یک کلاس با نام رنگ ایجاد می کنیم و در آن رنگ ماشین و وزن ماشین را تعریف می کنیم بعد یک کلاس دیگر با نام `door` تعریف می کنیم که در آن تعداد در ماشین و نام لاستیک را تعریف می کنیم .

برای ایجاد یک کلاس کافیست بر روی نام پروژه ای که ایجاد کردید کلیک راست [3] کرده و یک کلاس با نام دلخواه به برنامه اضافه کنید مثل شکل ۱-۱۹



بعد نام کلاس خود را به color تغییر دهید



بعد بر روی دکمه add کلیک کنید تا کلاس شما ایجاد شود

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace WindowsFormsApplication33
{
    class color
    {
    }
}
```

کلاس رنگ ایجاد شد حالا می توانیم هر چی بخواهیم داخل کلاس تعریف کنیم از جمله متغیرها و توابع  
مثال:

```
class color
{
    public string clr;
    public int Weigh;
}
```

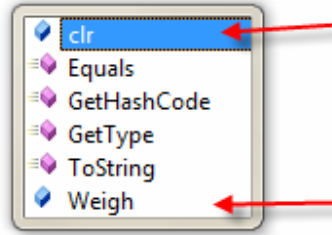
همانطور که ملاحظه کردید ما دو متغیر عمومی برای رنگ و وزن تعریف کردیم توجه داشته باشید اگر عمومی تعریف نمی کردیم نمی توانستیم به آنها در برنامه دسترسی داشته باشیم فقط داخل همین کلاس می توان به آنها دسترسی داشته باشیم .

مثال : حالا بر روی فرم برنامه یک دکمه قرار دهید و رویداد کلیک آن را فراخوانی کنید(روی آن دوبار کلیک کنید) بعد یک شی از کلاس رنگ ایجاد کنید نحوه کار به این صورت است که اول نام کلاس را می نویسیم بعد یک شی با نام دلخواه جلوی نام کلاس می نویسیم بعد عمل **new** کردن را انجام می دهیم مثل کد زیر:

```
private void button1_Click(object sender, EventArgs e)
{
    color mycolor = new color();
}
```

بعد با استفاده از شی **mycolor** می توانید به اعضای داخلی کلاس دسترسی داشته باشید مثال می توانید رنگ ماشین را زرد و وزن آن را ۱۰۰ کیلو مقدار دهی کنید کنید مثل:

```
color mycolor = new color();
mycolor.
```



```
private void button1_Click(object sender, EventArgs e)
{
    color mycolor = new color();
    mycolor.clr = "زرد";
    mycolor.Weigh = 100;
}
```

حال می توانید با استفاده از همان شی آنها در یک تکست باکس قرار دهید:

```
color mycolor = new color();
mycolor.clr = "زرد";
mycolor.Weigh = 100;
////////////////////
textBox1.Text = mycolor.clr;
textBox2.Text = mycolor.Weigh.ToString();
```

فرض که این مشخصات برای یک ماشین است اگر خواستید برای ماشین دیگر مشخصات دیگری انتخاب کنید

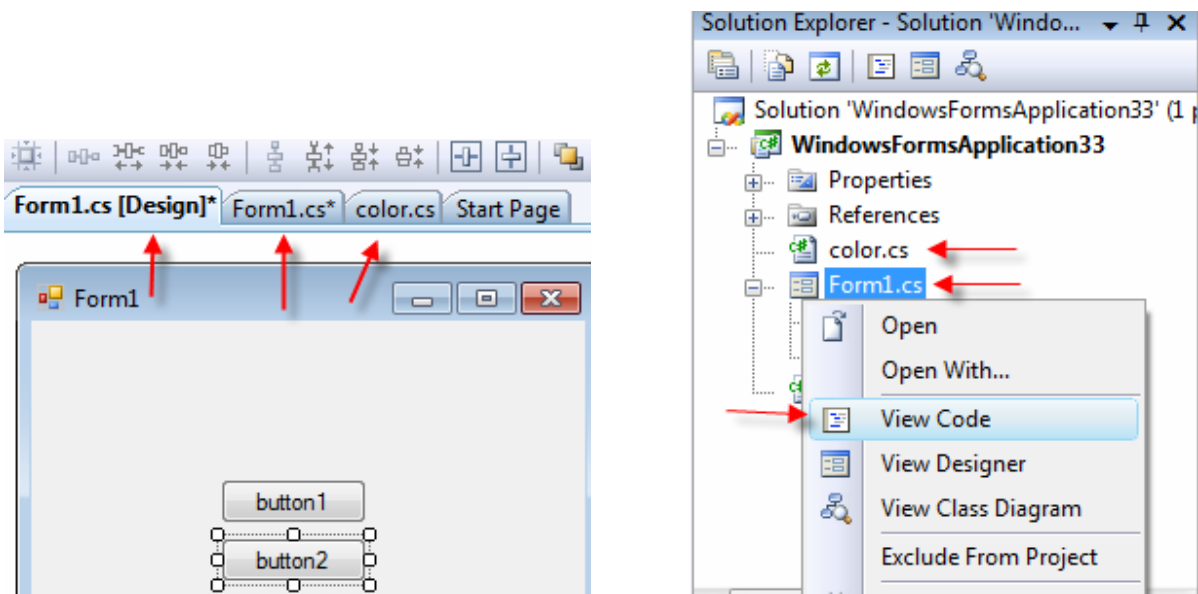
باید یک شی دیگری از این کلاس تعریف کنید!!! مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    color mycolor = new color();
    mycolor.clr = "زرد";
    mycolor.Weigh = 100;
    //////////////////////
    color mycolortwo = new color();
    mycolortwo.clr = "آبی";
    mycolortwo.Weigh = 500;
    //////////////////////
    textBox1.Text = mycolor.clr;
    textBox2.Text = mycolortwo.clr;
}
```





نکته : همانطور که قبلا گفتیم برای دسترسی به کلاس ها فرم ها (فرم ها هم کلاس هستند!!) می توانید از پنجره solution explorer به آنها دسترسی داشته باشید



نکته : حال یک دکمه دیگر بر روی فرم ایجاد کنید و رویداد کلیک آن را فراخوانی کنید در دکمه دوم نیز می توانید یک شی از آن ایجاد کنید ولی به شی قبلی که در دکمه اول ایجاد شد نمی توانید دسترسی داشته باشید برای دسترسی به شی قبلی باید کلاس را در خارج از بازه تعریف کنید تا همه بتوانند استفاده کنند به این نوع تعریف ,متغیر ها و شی های عمومی می گویند یعنی متغیر ها را نیز می توانید در خارج تعریف کنید تا همه بتوانند دسترسی داشته باشند وقتی ما متغیر و یا شی را داخل یک تابع (مثل رویداد کلیک دکمه) تعریف می کنیم دیگران نمی توانند از آن استفاده کنند که به این نوع تعریف محلی می گویند مثال:

```

InitializeComponent();
}
color mycolor = new color();
color mycolortwo = new color();
color my3 = new color();
int num;
private void button1_Click(object sender, EventArgs e)
{
    mycolor.clr = "زرد";
    mycolor.Weigh = 100;
    //////////////////////////////////////
    mycolortwo.clr = "آبی";
    mycolortwo.Weigh = 500;
    //////////////////////////////////////
    textBox1.Text = mycolor.clr;
    textBox2.Text = mycolortwo.clr;
    color mykh = new color();
    mykh.Weigh = 9;
}

private void button2_Click(object sender, EventArgs e)
{
    num = 78;
    my|
}

```

شي ها و متغير عمومي ←

شي محلي فقط داخل دکمه يک استفاده مي شود ←

همانطور که ملاحظه مي فرماييد فقط به عمومي ها دسترسي دارد

حال یک کلاس دیگر با نام door ایجاد می کنیم و در آن نیز متغیرهایی تعریف می کنیم برای ایجاد این کلاس مثل روش قبل عمل می کنیم فقط نام آن را door انتخاب می کنیم .

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace WindowsFormsApplication33
{
    class door
    {
        public int numberOfdoor;
        public string NTire;
    }
}

```

بعد از این کلاس نیز یک شی ایجاد کنید و از آن استفاده کنید مثال:

```
private void button2_Click(object sender, EventArgs e)
{
    door myclass = new door();
    myclass.NTire = "بارز";
}
```

یک تابع به کلاس خود اضافه کنید که جمع دو عدد را برای ما برگرداند مثل کد زیر:

```
class door
{
    public int numberOFdoor;
    public string NTire;
    public int MYT(int i, int f)
    {
        return (i + f);
    }
}
```

و کد زیر را برای دکمه بنویسید:

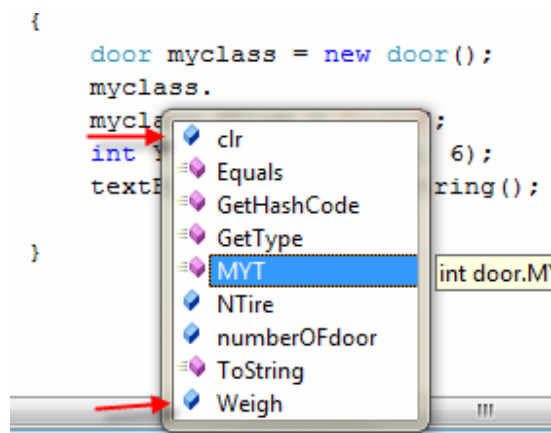
```
private void button2_Click(object sender, EventArgs e)
{
    door myclass = new door();
    myclass.NTire = "بارز";
    int Y = myclass.MYT(4, 6);
    textBox1.Text = Y.ToString();
}
```



ارث بری:

همانطور که ملاحظه می کنید هر کلاس فقط به اعضای داخلی خود می تواند دسترسی داشته باشد مگر اینکه یک کلاس را از کلاس دیگر ارث بریم به این ارث بری رابطه پدر و فرزند نیز گفته می شود در این مثال اگر کلاس door را از کلاس color به ارث بریم وقتی یک شی از door تعریف می کنیم با استفاده از آن شی می توانیم به اعضای داخلی کلاس color نیز دسترسی داشته باشیم ولی برعکس امکان پذیر نیست. نحوه ارث بری به این صورت است که جلوی نام کلاس علامت : گذاشته بعد نام کلاسی که می خواهیم از آن ارث بریم می نویسیم مثل:

```
namespace WindowsFormsApplication33
{
    class door : color
    {
        public int numberOFdoor;
        public string NTire;
        public int MYT(int i, int f)
        {
            return (i + f);
        }
    }
}
```



### سازنده کلاس :

همانطور که ملاحظه کردید وقتی یک شی از کلاس تعریف می کنیم در آخر از علامت ( ) استفاده می کنیم خوب اگر بخواهیم وقتی یک شی تعریف می کنیم همان موقع متغیر هایی را داخل کلاس مقدار دهی کنیم از سازنده ها استفاده می کنیم محل فرستادن مقادارها همان داخل پرانتز موقع تعریف شی است. برای تعریف یک سازنده کفایت در داخل کلاس مانند زیر عمل کنیم: (سازنده ها هم نام کلاس تعریف می شوند)

```
class color
{
    public color(int i, string j)
    {
        this.clr = j;
        this.Weigh = i;
    }
    public string clr;
    public int Weigh;
}
```

حال موقع تعریف یک شی از کلاس آن را نیز مقدار دهی می کنیم مثال:

قبل از نوشتن کد اگر ارث بری را ایجاد کردید بهتر است حذف کنید.

```
private void button2_Click(object sender, EventArgs e)
{
    color cl = new color(4, "قرمز");
    textBox1.Text = cl.clr;
}
```

حالا اگر خواستید موقع تعریف شی مثل قبل پرانتز خالی بزارید برنامه دچار اشکال می شود مگر اینکه یک سازنده دیگر تعریف کنید و آن را خالی بزارید مثل:

```
class color
{
    public color(int i, string j)
    {
        this.clr = j;
        this.Weigh = i;
    }
    public color()
    {
    }
    public string clr;
    public int Weigh;
}
```

توجه: از جمله مزایای دیگر کلاس ها کپسوله کردن است یعنی چندین نوع را می توانید در کنار یک دیگر قرار بدهید مثلا اگر خواستید مشخصات هزار دانشجو را که شامل نام, نام خانوادگی, شماره دانشجو, تلفن آدرس و دیگر مشخصات را نگه دارید باید چکار کنید یک راه اینست که چندین آرایه به طول هزار برای هر مشخصه تعریف کنید مثلا یک آرایه برای نام و یک آرایه برای نام خانوادگی و به همین ترتیب تعریف کنید و بعد یکی یکی آنها را مقدار دهی کنید که این اصلا روش خوبی نیست با استفاده از کلاس می توانید این کار را با یک آرایه انجام بدهید یعنی برای هر دانشجو یک شی تعریف می کنیم و در آرایه ای از **object** ها ذخیره می کنیم.

نکته: **object** نیز نوعی مانند **int** است که یک شی را در خود ذخیره می کند. البته این کار را می توانید در یک **ArrayList** ذخیره کنید که بعدا آن را توضیح خواهم داد.

مثال : برنامه مشخصات دانشجو را بنویسید ؟

برای اینکار یک برنامه جدید ایجاد کنید ۴ تکست باکس بر روی صفحه قرار دهید و ۴ لیبل در صورت علاقه در صفحه قرار دهید و یک دکمه ذخیره بر روی فرم قرار دهید مطابق شکل ۱-۲۰

حال یک کلاس با نام **stud** ایجاد کنید مثل کد زیر

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace WindowsFormsApplication34
{
    class stud
    {
        public string Name;
        public string Family;
        public int StudNumber;
        public int Tel;
    }
}
```

بر روی دکمه ذخیره دوبار کلیک کنید (رویداد کلیک) و یک آرایه به طول ۱۰۰۰ ایجاد کنید

```
private void button1_Click(object sender, EventArgs e)
{
    object[] MStud = new object[1000];
}
```

روش کار به این صورت است که با هر بار کلیک بر روی دکمه ذخیره مشخصات یک دانشجو در یک خانه آرایه ذخیره شود برای گردش در آرایه نیز از یک اندیس استفاده می کنیم تا در هر بار کلیک دکمه یک واحد به اندیس اضافه شود و در خانه بعدی آرایه ذخیره کنیم. مثل:

```
int i = 0;
private void button1_Click(object sender, EventArgs e)
{
    object[] MStud = new object[1000];
}
```

اندریس را خارج تعریف کردم تا هر وقت که دکمه را کلیک می کنیم مقدار اولیه از دست نرود و یک شی از کلاس تعریف می کنیم ولی **new** نمی کنیم مثل کد زیر:

```
int i = 0;
stud mystud;
private void button1_Click(object sender, EventArgs e)
{
    object[] MStud = new object[1000];
}
```

اینکه چرا بالا **new** نکردیم علت اینست که به چندین شی نیاز داریم. حالا در هر بار کلیک دکمه یک شی ایجاد می کنیم و آن شی را در آرایه ذخیره می کنیم. به این نکته توجه داشته باشید که شی که در هر بار ایجاد می شود متفاوت است با شی قبلی هر چند دارای نام یکسانی هستند.

```
int i = 0;
stud mystud;
object[] MStud = new object[1000];
private void button1_Click(object sender, EventArgs e)
{
    آرایه را به خارج انتقال دادم تا در دکمه های دیگر از آن استفاده کنم
    //////////////////////////////////////
    mystud = new stud();
    mystud.Name = textBox1.Text;
    mystud.Family = textBox2.Text;
    mystud.StudNumber = int.Parse(textBox3.Text);
    mystud.Tel = int.Parse(textBox4.Text);
    //////////////////////////////////////
    MStud[i] = mystud;
    i++;
}
```

حال یک دکمه دیگر قرار در صفحه قرار می دهیم و یک لیست باکس تا مقادیری ک ذخیره کردیم در لیست باکس نمایش دهیم

```
private void button2_Click(object sender, EventArgs e)
{
    for(int j=0;j<i;j++)
    {
        stud ii = (stud)MStud[j];
        listBox1.Items.Add(ii.Name + " " + ii.Family + " " +
ii.StudNumber + " " + ii.Tel);
    }
}
```

حال برنامه را اجرا کنید و دو مشخصه متفاوت ذخیره کنید در آخر هم دکمه نمایش را بزنید تا مشخصات را مشاهده کنید فقط به این نکته توجه داشته باشید که در هر مرحله دکمه ذخیره را بزنید



در ادامه می خواهیم یک جستجو نیز به برنامه اضافه کنیم که بر اساس شماره دانشجویی جستجو کند و در صورت پیدا کردن پیغام دهد که پیدا شده است برای این کار یک دکمه و یک تکست باکس دیگر اضافه کنید و برای دکمه سوم کد زیر را بنویسید روش کار به این صورت است که بر روی تک تک عناصر آرایه تا اندیسی که اضافه شده حرکت می کند و هر کجا برابر بود پیغام می دهد

```
private void button3_Click(object sender, EventArgs e)
{
    for (int j = 0; j < i; j++)
    {
        stud ii = (stud)MStud[j];
        if (ii.StudNumber == int.Parse(textBox5.Text))
        {
            textBox1.Text = ii.Name;
            textBox2.Text = ii.Family;
        }
    }
}
```



```

        textBox3.Text = ii.StudNumber.ToString();
        textBox4.Text = ii.Tel.ToString();
        MessageBox.Show("شد پیدا");
        break;
    }
}
}

```



## Get{}Set{}

اگر بخواهیم متغیرهای خصوصی را در کلاس مقدار دهی کنیم از **set** و اگر بخواهیم آن را از کلاس بخوانیم از **get** استفاده می کنیم . همچنین می توانیم موقع مقدار دهی آن را محدود کنیم مثلا اعداد بین یک تا پنج را فقط می توانیم در متغیر خصوصی ذخیره کنیم . برای مثال یک کلاس جدید ایجاد کنید و یک متغیر خصوصی داخل آن به صورت زیر تعریف کنید:

```

public class myclass
{
    private int _NUM;
    public int MYNUM
    {
        get
        {
            return _NUM;
        }
    }
}

```

```

    }
    set
    {
        _NUM = value;
    }
}

```

در مثال بالا متغیر خصوصی `_NUM` را با استفاده از `MYNUM` مقدار دهی و می خوانیم. (`value` همان مقداری است که در متغیر `MYNUM` قرار است قرار دهید).

```

private void button1_Click(object sender, EventArgs e)
{
    myclass mc = new myclass();
    mc.MYNUM = 1394;
    textBox1.Text = mc.MYNUM.ToString();
}

```

اگر خواستید می توانید موقع مقدار دهی محدود کنید :

```

set
{
    if (value > 0 && value < 15)
    {
        _NUM = value;
    }
}

```

اگر بخواهید فقط یک مقدار خصوصی را از کلاس بخوانید ولی نتوانید آن را تغییر دهید باید فقط از `get` استفاده کنید . مثال :

```

public class myclass
{
    private int _NUM;
    private string _Str;
    public string Str
    {
        get
        {
            return this._Str;
        }
    }
}

```

و اگر خواستید فقط یک مقدار بگیرید و در کلاس از آن استفاده کنید باید از `set` به تنهایی استفاده کنید مثال :

```

public string Str
{
    set

```

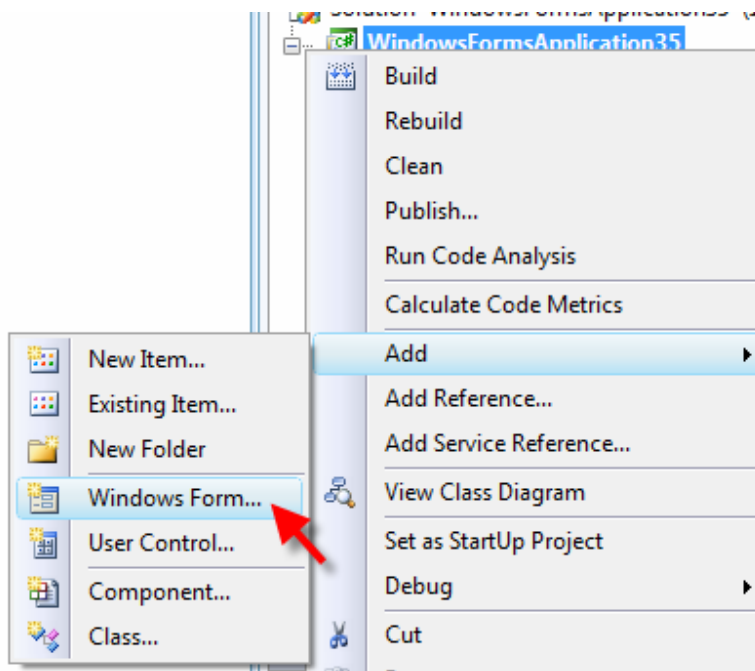
```

    {
        this.Str = value;
    }
}

```

## ۱-۱۰ استفاده از چندین فرم :

شاید در بعضی از برنامه ها خواستید از چندین فرم استفاده کنید مثلا یک دکمه **next** در صفحه اول قرار بدهید و با کلیک بر روی آن صفحه دوم فراخوانی شود . برای اضافه کردن فرم به پروژه مثل اضافه کردن پروژه عمل می کنید ولی اینبار بر روی **Windows Form** کلیک کنید و یک نام برای فرم خود انتخاب کنید و بروی دکمه **Add** کلیک کنید.



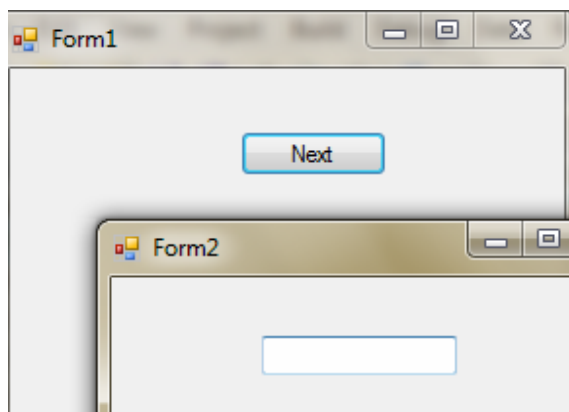
بر صفحه اول یک دکمه قرار داده و تکست آن را به **next** تغییر دهید و بروی فرم دوم هم یک تکست باکس قرار دهید و کد زیر را برای دکمه صفحه اول بنویسید

```

private void button1_Click(object sender, EventArgs e)
{
    Form2 m = new Form2();
    m.Show();
}

```

روش کار به این صورت است که چون فرمها جز کلاس ها هستند بنابراین برای دسترسی به آن ها یک شی از آنها ایجاد می کنیم و تابع `Show()` کلاس فرم را فراخوانی می کنیم .

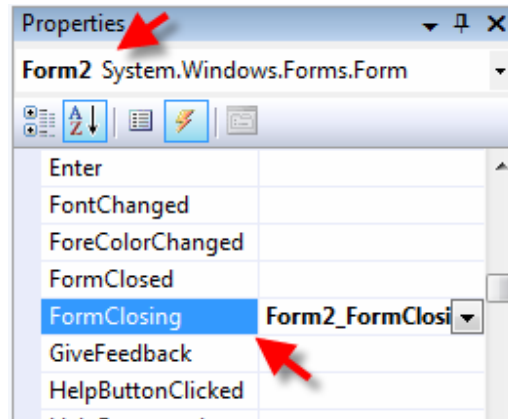


اگر خواستید فرم اول دیگر نمایش داده نشود باید آن را توسط تابع `ActiveForm.Hide()` پنهان کنید مثل کد زیر:

```
private void button1_Click(object sender, EventArgs e)
{
    Form1.ActiveForm.Hide();
    Form2 m = new Form2();
    m.Show();
}
```

و اگر خواستید به فرم اول باز گردید باید یک شی از آن ایجاد کنید و متد `show` آن را فراخوانی کنید .

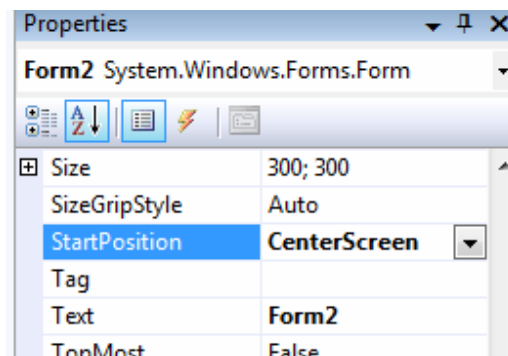
نکته: وقتی فرم اول را پنهان می کنید و فرم دوم در حال نمایش است دیگر وقتی فرم دوم را ببندید برنامه به طور کامل بسته نشده است بلکه فرم اول در حال اجرا است برای قطع فرم اول هم باید کد زیر را در قسمت `FormClosing` فرم دوم بنویسید برای فراخوانی این رویداد از قسمت رویداد فرم دوم بر روی `FormClosing` کلیک کنید.



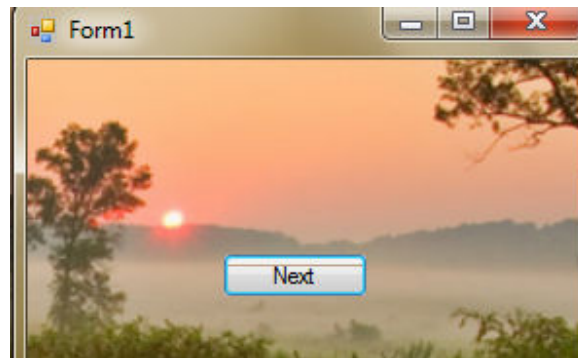
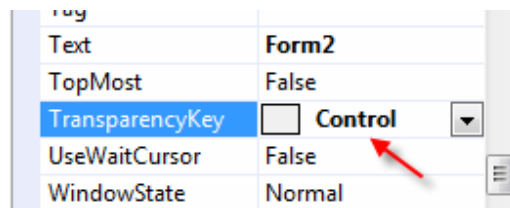
```
private void Form2_FormClosing(object sender, FormClosingEventArgs e)
{
    Application.Exit();
}
```

FormClosing زمانی اجرا می شود که کاربر در حال بستن فرم برنامه است . و تابع بالا هم باعث باعث بسته شده کل برنامه از ریشه می شود .

نکته : برای اینکه فرم برنامه شما هنگام اجرا در وسط صفحه نمایش قرار گیرد باید از قسمت properties و startPosition فرم را به CenterScreen تغییر دهید و اگر خواستید کاربر نتواند اندازه فرم را تغییر دهد باید maximizeBox را برابر False قرار دهید و formborderstyle را برابر FixedSingle قرار دهید.



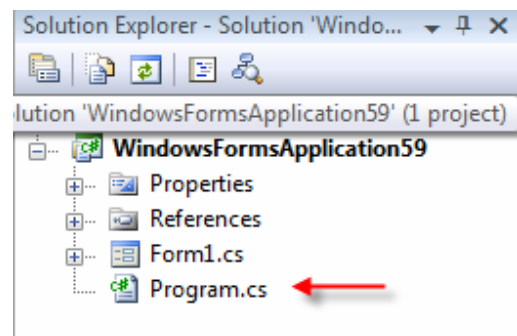
اگر خواستید بدنه فرم نمایش داده نشود و فقط ابزار های روی فرم نمایش داده شود باید TransparencyKey فرم را برابر رنگ Control قرار دهید.



عکس دستکتاپ ویا هرچیزی که پشت سر فرم است نمایش داده می شود.

نکته : اگر خواستید هر یک از فرمهای برنامه را به دلخواه به عنوان اولین صفحه (پدر) انتخاب کنید و موقع اجرای برنامه این صفحه به عنوان اولین صفحه اجرا شود باید از قسمت Program.cs نام کلاس (فرم) را به دلخواه تغییر دهید :

```
static void Main()
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderDefault();
    Application.Run(new Form1());
}
```



نکته :

اگر خواستید برنامه اجرایی شما فقط یک بار در حال اجرا باشد باید از Mutex به شکل کد زیر در Program استفاده کنید [1]:

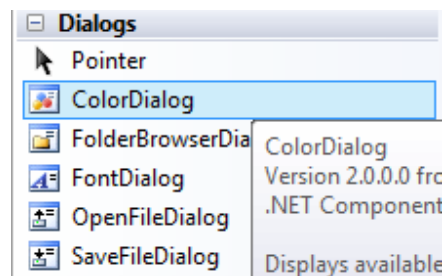
```
static void Main()
{
    bool boooooool;
```



موقع نیاز به تغییر مقادیر بود فقط کافیست آن مقدار ثابت را تغییر دهیم برای تعریف ثابت ها کافیست از علامت const در مقابل متغیرمان استفاده کنیم از جمله مزایای ثابت ها غیر قابل تغییر بودن آن در متن برنامه است. ولی می توانید آن را با مقدار دیگری جمع ببندید مثال :

```
const int price = 1200000;
private void button2_Click(object sender, EventArgs e)
{
    int hamed=price+2000;
    int nader = price * 2;
    int mahmod = price - 5000;
}
```

## ۱۱-۱ کار با ابزار Dialogs

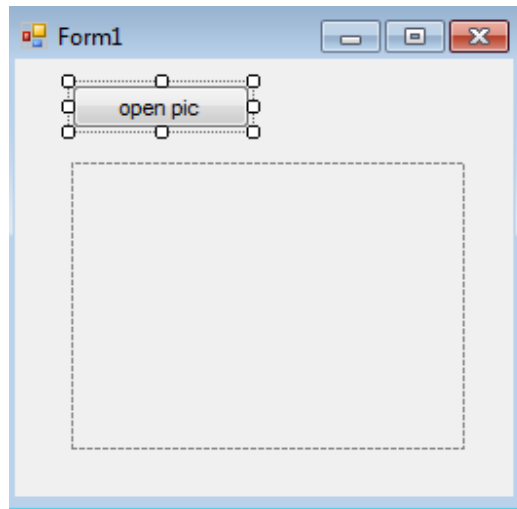


شاید برای شما بیشتر پیش آمده باشد که وقتی می خواهید برنامه ای را باز و یا ذخیره کنید صفحه ای برای شما نمایش داده می شود و از شما نام و مسیر فایل پرسیده می شود اینها همان dialogs هایی هستند که در این قسمت می خواهیم به آنها پردازیم.

مثال : برنامه این بنویسید که یک عکس را توسط openFileDialog نمایش دهد؟

برای انجام این برنامه یک pictureBox و یک openFileDialog و یک دکمه در فرم قرار دهید



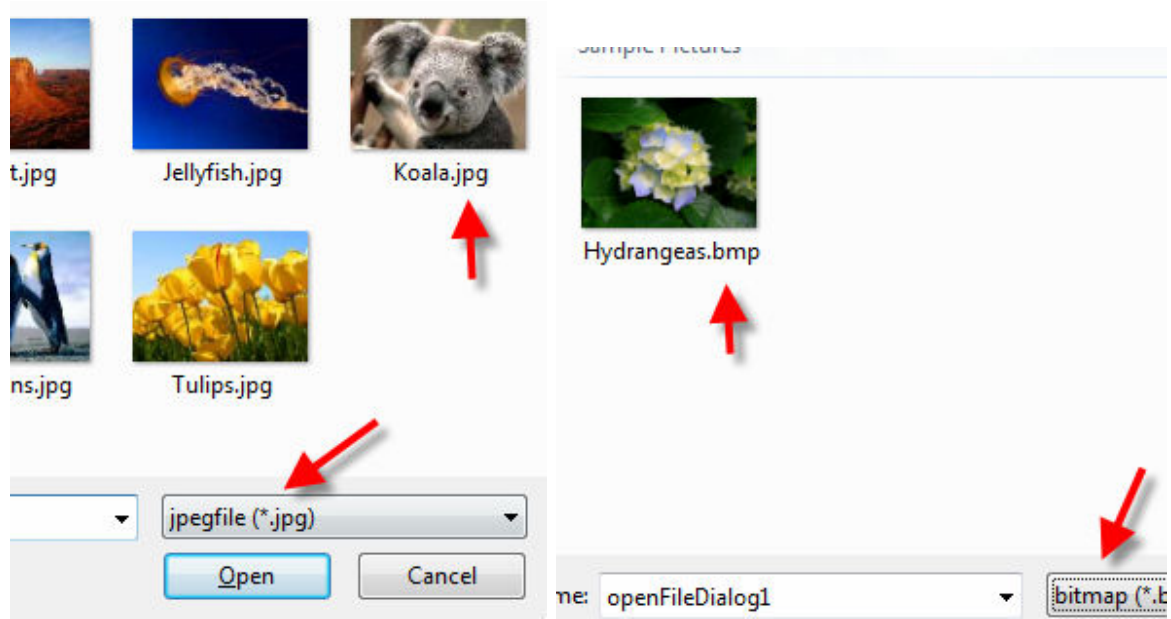


حال این کد را برای دکمه بنویسید. روش کار به این صورت است که تابع `ShowDialog()` باعث نمایش پنجره می شود و وقتی کاربر در پنجره فایلی را انتخاب می کند نام و مسیر کامل این فایل در `FileName` ذخیره می شود

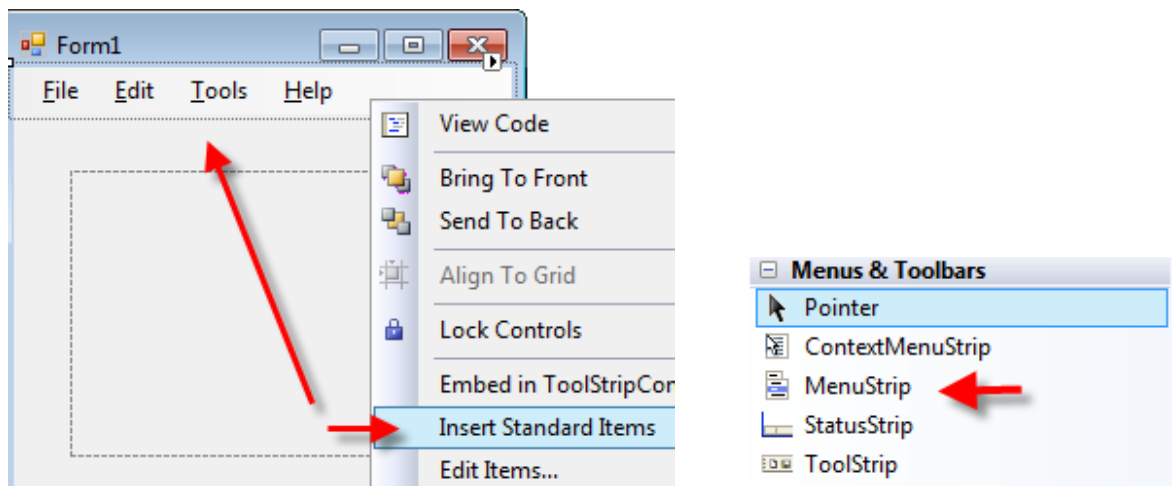
```
private void button1_Click(object sender, EventArgs e)
{
    openFileDialog1.ShowDialog();
    // pictureBox1.BackgroundImageLayout = ImageLayout.Stretch;
    pictureBox1.BackgroundImage =
Image.FromFile(openFileDialog1.FileName);
}
```

اگر خواستید `openfiledialog` را فیلتر کنید مثلاً فقط عکس هایی با پسوند `jpeg` و `bmp` باز کند (نمایش دهد) از `Filter` به صورت کد زیر استفاده می کنیم :

```
private void button1_Click(object sender, EventArgs e)
{
    openFileDialog1.Filter = "jpegfile (*.jpg)|*.jpg|bitmap|*.bmp|
All files (*.*)|*.*";
    openFileDialog1.ShowDialog();
    // pictureBox1.BackgroundImageLayout = ImageLayout.Stretch;
    pictureBox1.BackgroundImage =
Image.FromFile(openFileDialog1.FileName);
}
```



حال برنامه را تغییر می دهیم و به جای دکمه یک MenuStrip در صفحه قرار می دهیم



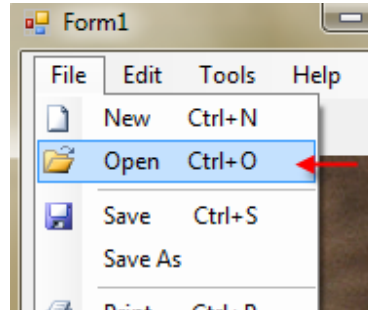
حال از File بر روی Open دو بار کلیک کنید و بعد کد زیر را داخل آن بنویسید

```
private void openToolStripMenuItem_Click(object sender, EventArgs e)
{
    openFileDialog1.Filter = "jpegfile (*.jpg)|*.jpg|bitmap|*.bmp|
All files (*.*)|*.*";
    openFileDialog1.ShowDialog();
    // pictureBox1.BackgroundImageLayout = ImageLayout.Stretch;
}
```

```

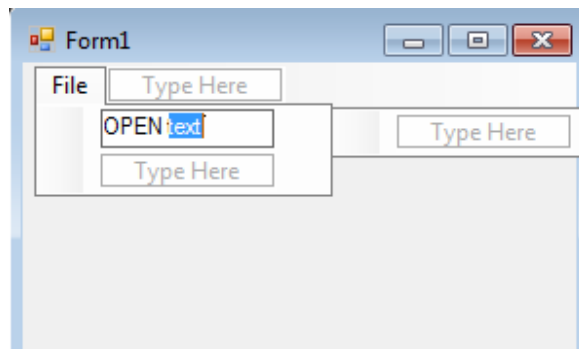
pictureBox1.BackgroundImage =
Image.FromFile(openFileDialog1.FileName);
}

```



مثال : برنامه ای که یک فایل متنی txt را باز می کند و در تکست باکس نمایش میدهد

مثل مثال قبلی یک منو بر روی فرم قرار دهید و منو را مطابق شکل تنظیم کنید



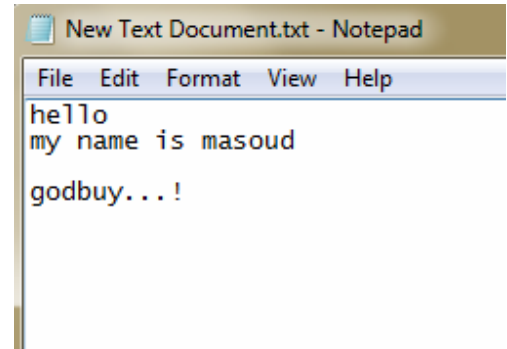
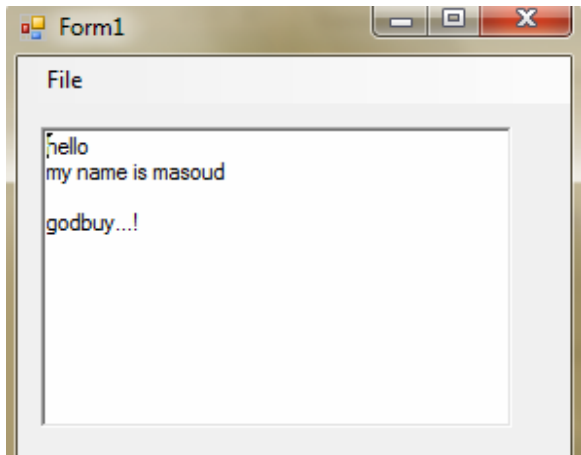
و یک richTextBox و یا textbox بر روی فرم قرار دهید و اندازه آن را تغییر دهید تا سایش بزرگ شود حالا بر روی open text درمنودو بار کلیک کنید و کد زیر را برای آن بنویسید

```

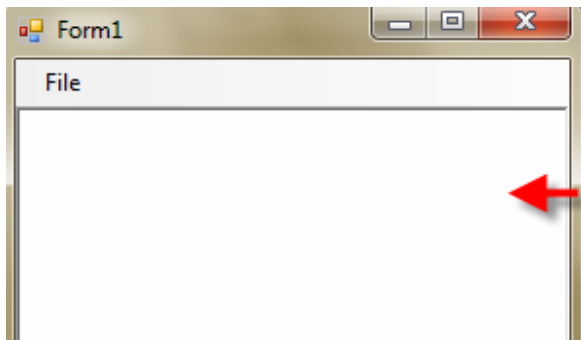
private void oPENTextToolStripMenuItem_Click(object sender, EventArgs
e)
{
    openFileDialog1.Filter = "txt files (*.txt)|*.txt|All files
(*.*)|*.*";
    openFileDialog1.ShowDialog();
    string str =
System.IO.File.ReadAllText(openFileDialog1.FileName);
    richTextBox1.Text = str;
}

```

حال یک فایل متنی در جایی ایجاد کنید و عبارت دلخواه خود را در آن بنویسید و آن را ذخیره کنید و بعد برنامه را اجرا کنید و opentext را بزنید و آن فایل متنی را انتخاب کنید .



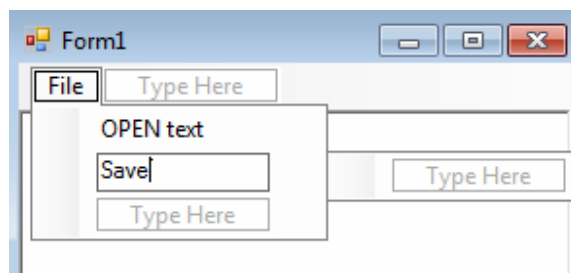
اگر خواستید dock richTextBox1 را برابر Fill قرار دهید تا تمام صفحه را در برگیرد.



Cursor	IBeam
DetectUrls	True
Dock	Fill
EnableAutoDragDrop	False
Enabled	True
Font	Microsoft Sans Serif

مثال:

اگر خواستید متن داخل ریچ باکس را ذخیره کنید به جای openfiledialog یک savefiledialog در صفحه قرار دهید و این کد را بنویسید



```
private void saveToolStripMenuItem_Click(object sender, EventArgs e)
{
    saveFileDialog1.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
    saveFileDialog1.ShowDialog();
}
```

```

        System.IO.File.WriteAllText(saveFileDialog1.FileName,
richTextBox1.Text);
    }

```

## ColorDialog

از این دیالوگ برای انتخاب رنگ استفاده می شود

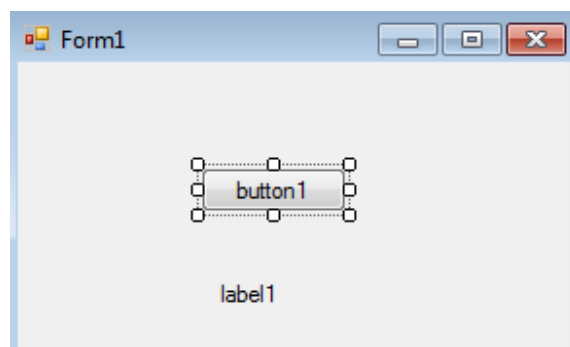
## FontDialog

از دیالوگ برای انتخاب فونت استفاده می شود.

مثال :

برنامه ای که رنگ پشت زمینه فرم و فونت label را تغییر می دهد

یک لیبل و دکمه بر روی فرم قرار دهید و دو تا دیالوگ فونت و رنگ را نیز بر روی فرم قرار دهید



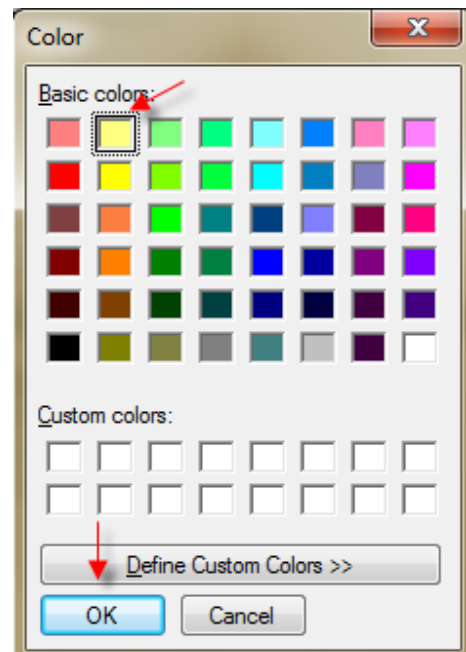
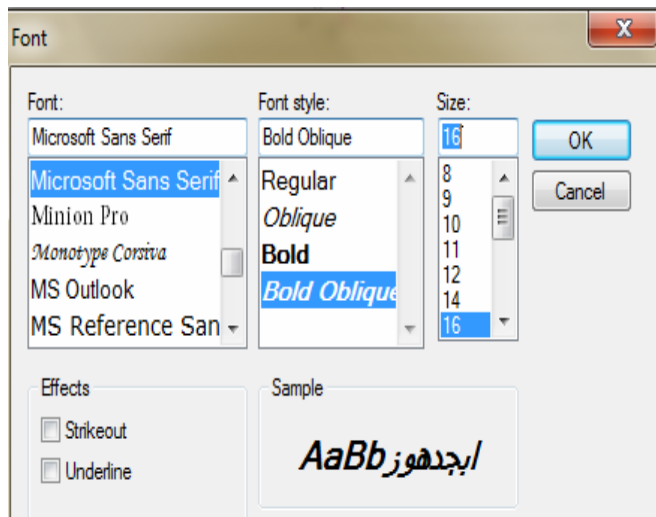
بر روی دکمه کلیک کنید و کد زیر را برای آن بنویسید

```

private void button1_Click(object sender, EventArgs e)
{
    colorDialog1.ShowDialog();
    fontDialog1.ShowDialog();
    this.BackColor = colorDialog1.Color;
    label1.Font = fontDialog1.Font;
}

```

حال برنامه را اجرا کنید و یک رنگ و فونت برای خود انتخاب کنید تا نتیجه را مشاهده فرمایید.



حال کد قبلی را به شکل زیر تغییر دهید و نتیجه را مشاهده فرمایید:

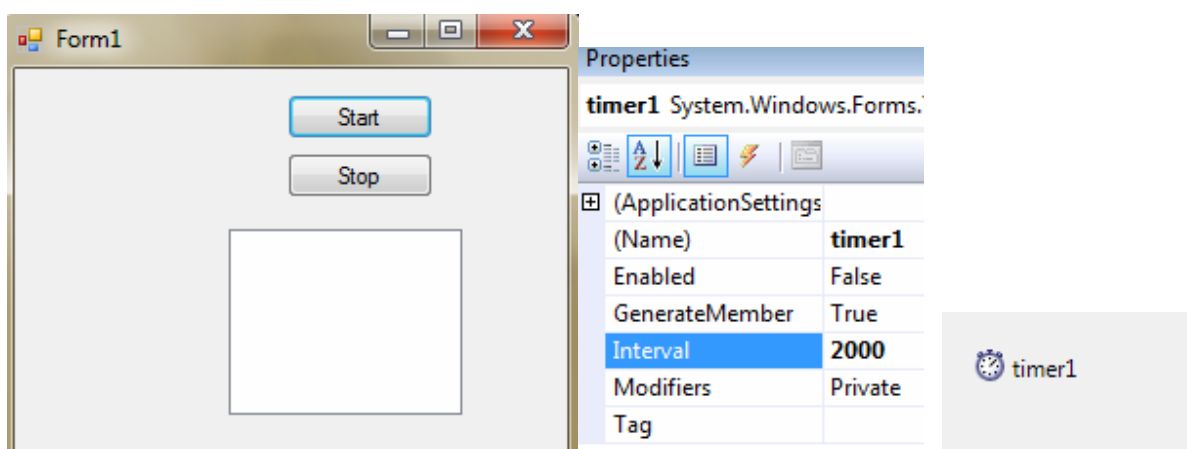
```
private void button1_Click(object sender, EventArgs e)
{
    colorDialog1.ShowDialog();
    fontDialog1.ShowDialog();
    button1.BackColor = colorDialog1.Color;
    button1.Font = fontDialog1.Font;
}
```



با بقیه دیالوگ ها در ادامه و در داخل برنامه ها آشنا خواهیم شد.

## Timer 1-12

اگر خواستید برنامه ای بنویسید که در مدت زمان خاصی اجرا بشود از تایمر استفاده می شود که کار با آن بسیار ساده است از تایمر در بسیاری از برنامه ها استفاده می شود. اساس کار بسیاری از برنامه ها با تایمر است و اگر تایمر وجود نداشت نوشتن این برنامه ها امکان پذیر نبود. مثال : برنامه ای بنویسید که در هر دو ثانیه یک عنصر به لیست باکس اضافه کند؟ برای انجام این کار یک تایمر و یک لیست باکس و دو دکمه در صفحه قرار دهید و **interval** تایمر را به ۲۰۰۰ تغییر دهید (هر هزار برابر یک ثانیه است) بعد روی تایمر دو بار کلیک کنید تا رویداد آن فراخوانی شود بعد کد زیر را داخل آن



بنویسید :

```
private void timer1_Tick(object sender, EventArgs e)
{
    listBox1.Items.Add("Computer");
}
```

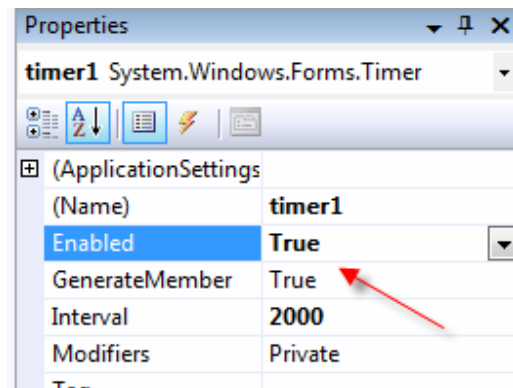
بعد به ترتیب کد های زیر را برای دکمه اول و دوم بنویسید:

```
private void button1_Click(object sender, EventArgs e)
{
    timer1.Start();
    // timer1.Enabled = true;
}

private void button2_Click(object sender, EventArgs e)
{
    timer1.Stop();
}
```

بعد برنامه را اجرا کنید و بر روی دکمه استارت کلیک کنید تا تایمر شروع به کار کند و هر وقت خواستید آن را متوقف کنید.

اگرخواستید بدون دکمه و خودکار تایمر شروع به کار کند می توانید همان اول کار `Enabled` تایمر را `true` کنید



یا بر روی فضای خالی فرم کلیک کنید تا متد لود فرم فراخوانی شود و کد زیر (فعالسازی) را داخل آن بنویسید.

```
private void Form1_Load(object sender, EventArgs e)
{
    timer1.Start();
}
```

همچنین می توانید `Interval` تایمر را نیز از طریق کد نویسی تغییر دهید مثل کد زیر:

```
private void Form1_Load(object sender, EventArgs e)
{
    timer1.Interval = 100;
    timer1.Start();
}
```

برخی از امکانات سی شارپ مربوط می شود به توابع کتابخانه ای که دربخش قبل مختصری به آن اشاره کردیم و در اینجا میخواهیم کمی بیشتر درباره آنها و کلاس های داخلی آنها بحث کنیم . گفتیم هدر هایی که بیشتر مورد استفاده قرار می گیرند به برنامه به صورت پیش فرض اضافه شده اند در اینجا به نحوه استفاده از آنها می پردازیم



. مثلا در هدر زیر که به برنامه اضافه شده است (شما اضافه کنید!!) کلاس های بسیار خوبی وجود دارد که به چند تای آن اشاره می کنیم.

```
using System.Collections;
```

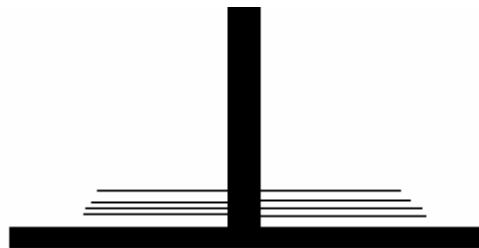
مثلا کلاس پشته ای [3] در این سیستم وجود دارد که کار پشته را انجام می دهد برای استفاده از آن کافیست یک شی از آن ایجاد کنیم مثل کد زیر

```
System.Collections.Stack Stk = new System.Collections.Stack;
```

حالا با استفاده از شی مورد نظر می توانید به پشته عنصر مورد نظر را اضافه کنید و یا از آن بردارید. دو تابعی که در این کلاس وجود دارد عملیات اضافه کردن به پشته و یا برداشتن از بالای پشته را انجام میدهند.

```
Stk.Push1;  
Stk.Push2;  
int i = Stk.Pop;
```

پشته مانند ظرفی عمل می کند که وقتی عنصری به آن اضافه شود می رود و در ته ظرف قرار می گیرد و عنصر بعدی روی عنصر اول قرار می گیرد و به همین ترتیب عناصر دیگر قرار می گیرند و وقتی می خواهیم عنصری از پشته برداریم آن عنصر آخرین عنصری است که به پشته اضافه کردیم. (مانند قاب CD)



در ضمن باید به این نکته نیز اشاره کنم که در صورتی که هدر به برنامه اضافه شده باشد لازم نیست قسمت اول کد را بنویسید یعنی اگر فقط نام کلاس را بنویسید و یک شی از آن ایجاد کنید کافی است مانند

```
Stack Stk = new Stack;
```

از دیگر کلاس های این کتابخانه می توان به لیست های پیوندی اشاره کرد که در کلاس `ArrayList` پیاده سازی شده است برای استفاده از این کلاس نیز کفایت از آن یک شی ایجاد کنیم مانند

```
ArrayList myarray = new ArrayList;
```

و به آن مقدار مورد نظر را اضافه کنیم

```
myarray.Add(123);
```

به روش زیر نیز می توانیم آن را مقدار دهی کنیم

```
myarray[0]=123;
```

از ویژگی ها جالب این کلاس این است که لازم نیست مانند آرایه ها طولش را ذکر کنیم بلکه هر وقت عنصری به آن اضافه می کنیم خودش فضا برای آن در نظر می گیرد در واقع این کلاس مانند لیست پیوندی عمل می کند.

اگر یادتان باشد در بحث کلاسها مثالی را حل کردیم که مشخصات دانشجویان را در آرایه ای از `object` ذخیره می کرد حالا همان مثال را با استفاده از `ArrayList` حل می کنیم .

```
using System.Collections;
namespace WindowsFormsApplication34
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        stud mystud;
        ArrayList Mstud = new ArrayList();
        private void button1_Click(object sender, EventArgs e)
        {
            //////////////////////////////////////
            mystud = new stud();
            mystud.Name = textBox1.Text;
            mystud.Family = textBox2.Text;
            mystud.StudNumber = int.Parse(textBox3.Text);
            mystud.Tel = int.Parse(textBox4.Text);
            //////////////////////////////////////
            Mstud.Add(mystud);
        }
        private void button2_Click(object sender, EventArgs e)
```

```

{
    for (int j = 0; j < Mstud.Count; j++)
    {
        stud ii = (stud)Mstud[j];
        listBox1.Items.Add(ii.Name + " " + ii.Family + " " +
ii.StudNumber + " " + ii.Tel);
    }
}

private void button3_Click(object sender, EventArgs e)
{
    for (int j = 0; j < Mstud.Count; j++)
    {
        stud ii = (stud)Mstud[j];
        if (ii.StudNumber == int.Parse(textBox5.Text))
        {
            textBox1.Text = ii.Name;
            textBox2.Text = ii.Family;
            textBox3.Text = ii.StudNumber.ToString();
            textBox4.Text = ii.Tel.ToString();
            MessageBox.Show("شد پیدا");
            break;
        }
    }
}
}
}
}

```

کد دکمه جستجو button3\_Click را می توانید به این شکل نیز تغییر دهید یعنی از حلقه FOREACH استفاده کنید.

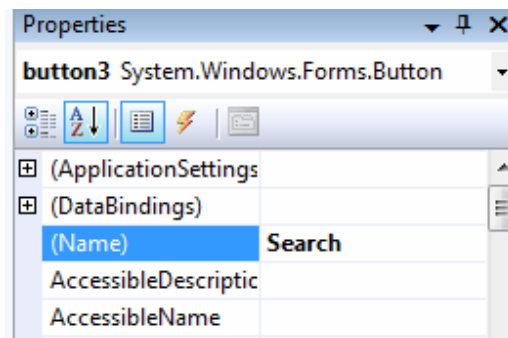
```

private void button3_Click(object sender, EventArgs e)
{
    foreach (stud ii in Mstud)
    {
        if (ii.StudNumber == int.Parse(textBox5.Text))
        {
            textBox1.Text = ii.Name;
            textBox2.Text = ii.Family;
            textBox3.Text = ii.StudNumber.ToString();
            textBox4.Text = ii.Tel.ToString();
            MessageBox.Show("شد پیدا");
            break;
        }
    }
}

```

همانطور که ملاحظه کردید به آسانی و با استفاده از **arraylist** این کار انجام می گیرد.

نکته : برای اینکه با کنترل ها به آسانی کار کنید بهتر است ( Name ) آن را تغییر دهید برای مثال اگر قبل از کد نویسی این برنامه برای دکمه سه button3 نام آن را به Search تغییر دهید در قسمت کد نویسی به آسانی می توانید به آنها دسترسی داشته باشید.

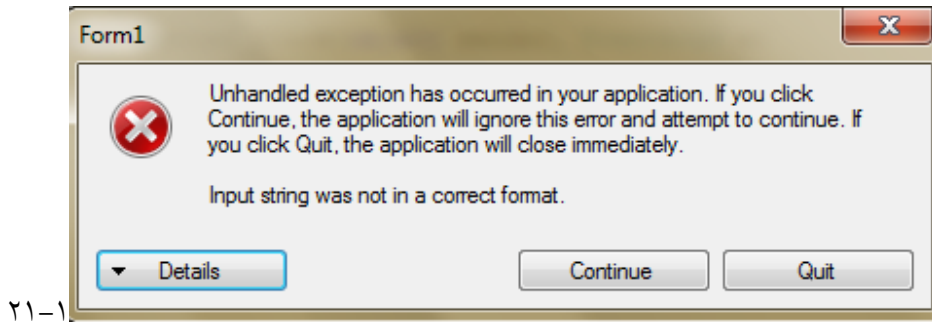


```
private void Search_Click(object sender, EventArgs e)
{
}

```

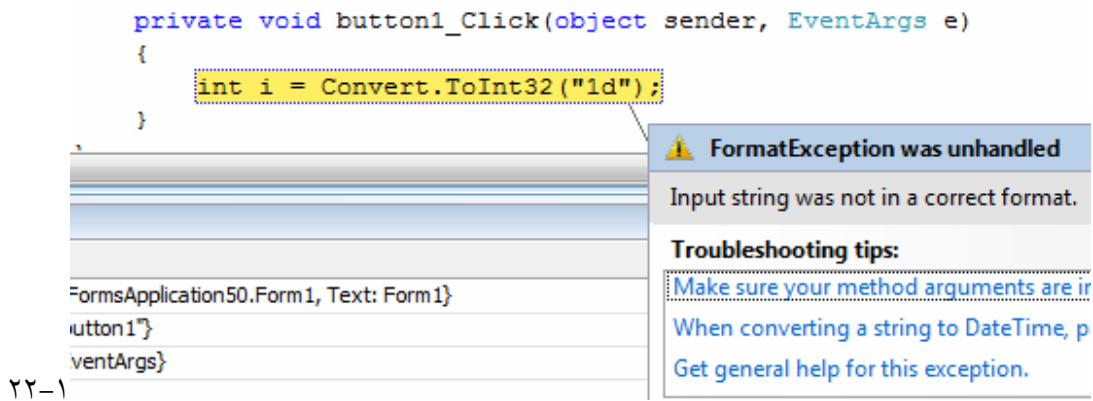
### ۱۳-۱ کنترل استثناها :

در طول برنامه هایی که نوشتیم شاید برای شما خیلی پیش آمده که برنامهتون با ایراد رو برو بشه و پنجره خطاری برای شما نمایش داده شود مثل پنجره خطای شکل ۱-۲۱



۲۱-۱

یا اگر برنامه را با F5 اجرا کردید سی شارپ به شما اخطار و توضیحاتی می دهد مثل شکل ۲۲-۱



۲۲-۱

که به ما می گوید رشته ای که شما وارد کردید درست نیست مثلاً برنامه ای نوشتید که یک عدد را از تکست باکس دریافت می کند ولی چون تکست باکس ما نوعی که دریافت می کند از نوع استرینگ است بنابراین باید عمل تبدیل به `int` را توسط تابعی انجام بدهیم ولی این تابع نیاز به استرینگ صحیح از نوع اعداد است مانند "1234" و اگر کاربر این رشته را وارد کند "123m" برنامه دچار اشکال می شود در واقع می گوییم دچار استثنا شده است در اینجا می خواهیم ببینیم چه راه حلهایی می توانیم پیشنهاد کنیم تا این مشکل حل شود یک راه حل اینست که با استفاده از شرط ها یا دیگر امکانات این مشکل را کنترل کنیم مثلاً اگر بخواهیم کاربر در تکست باکس رشته خالی وارد نکند با یک شرط می توانیم این مشکل را کنترل کنیم مثل کد زیر:

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox1.Text != "")
    {
        int i = Convert.ToInt32(textBox1.Text);
    }
}
```

و یا بخواهیم که کاربر فقط عدد وارد کند در این صورت نمی تواند رشته نادرست به غیر از عدد وارد کند مثل کد زیر:

```
private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsNumber(e.KeyChar))
    {
        e.Handled = true;
    }
}
```

ولی این روشها خوب نیستند چون در بعضی کد نویسی ها ممکن است ما نتوانیم بعضی از اشکالات را که ممکن است رخ دهد را پیش بینی و حتی روشی برای آن پیدا کنیم بنابراین باید به سراغ روش کلی دیگری برویم که آن روش کنترل استثناها با `try{} catch {}` است. روش کار به این صورت است که ما کدهایمان را داخل `try` می نویسیم و اگر برنامه دچار استثنا شد از همان خط اجرای برنامه به داخل `catch` انتقال میابد و کدهای داخل `catch` اجرا می شوند و در آنجا باید کد دیگری بنویسیم که حتما می دانیم هیچ استثنایی در آن رخ نمی دهد مثل پیغام های خطا و یا اصلا هیچ کدی ننویسیم تا برنامه تمام شود.

برای مثال برنامه ای می نویسیم که اگر کاربر رشته نادرست وارد کرد دیگر برنامه دچار استثنا نشده و پیغامی به کاربر نمایش داده شود.

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        int i = Convert.ToInt32(textBox1.Text);
    }
    catch
    {
        MessageBox.Show("please enter a number!!!?");
    }
}
```

اگر خواستید می توانید داخل `catch` هیچ کدی ننویسید مثل کد زیر:

```
try
{
    int i = Convert.ToInt32(textBox1.Text);
}
catch
{
```

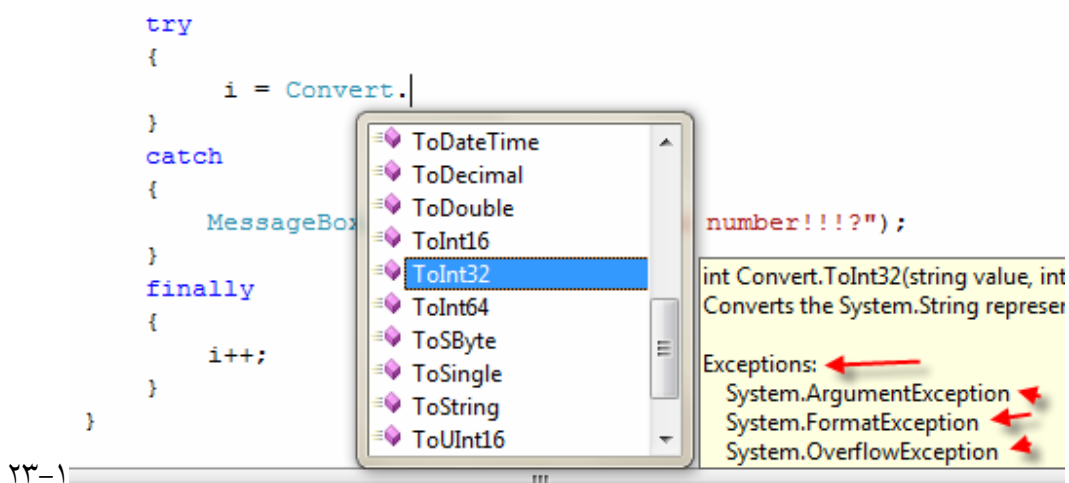
}

اگر خواستید چه استثنا رخ بدهد و یا ندهد کدی برای ما اجرا شود از `finally` استفاده می کنیم ولی باید کد داخل آن دارای استثنا نباشد مثال :

```
private void button1_Click(object sender, EventArgs e)
{
    int i = 0;
    try
    {
        i = Convert.ToInt32(textBox1.Text);
    }
    catch
    {
        MessageBox.Show("please enter a number!!!?");
    }
    finally
    {
        i++;
    }
}
```

اگر خواستید پیغامی از طرف سیستم به شما نمایش داده شود به این شکل عمل می کنیم (البته هر متد دارای پیغام استثنای منحصر به خود است برای فهمیدن پیغام استثنای آنها می توانید موقع نوشتن کد آنها را بفهمید

مثل شکل ۲۳-۱



حال که اسم اینها را فهمیدیم به این شکل عمل می کنیم :

```
private void button1_Click(object sender, EventArgs e)
```





از این کتابخانه می توان برای انجام عملیات گرافیکی استفاده کرد مثلاً برای ساختن یک عکس با استفاده از کلاس **bitmap** به فرمت های مختلف از این کلاس استفاده می شود. کلاس **bitmap** دارای دو تابع مهم با نامهای **GetPixel** و **SetPixel** است که وظیفه اولی به دست آوردن رنگ یک پیکسل است برای اینکار کفایت اندیس آن پیکسل را در عکس مشخص کنیم تا رنگ آن را برگرداند

```

bmp.GetPixel (|
Color Bitmap.GetPixel (int x, int y)
x: The x-coordinate of the pixel to retrieve.

```

و وظیفه دومی رنگ کردن پیکسل مورد نظر است برای اینکار باز کفایت اندیس آن پیکسل را مشخص کنیم و بعد رنگ را نیز ذکر کنیم تا آن پیکسل به رنگ مورد نظر درآید.

```

bmp.SetPixel (|
void Bitmap.SetPixel (int x, int y, Color color)
x: The x-coordinate of the pixel to set.

```

کلاس **Color** یک رنگ را درخود نگه می دارد و می تواند سه رنگ اصلی آن را نمایش دهد و یا بر اساس سه رنگ اصلی یک رنگ برای ما تولید کند برای مثال درکد زیر یک نوع از **Color** تعریف می کنیم و آن را با رنگ قرمز مقدار دهی می کنیم

```

Color Mycolor = Color.Red;

```

معادل کد بالا به شکل زیر است :

```

Color Mycolor = Color.FromArgb(255, 0, 0);

```

می توانید به سه رنگ اصلی که شامل RGB است دسترسی داشته باشید برای مثال درکد زیر سه رنگ اصلی رنگ سبز را نمایش دادیم :

```

Color Clr = Color.Green;
byte _R = Clr.R;
byte _G = Clr.G;
byte _B = Clr.B;

```

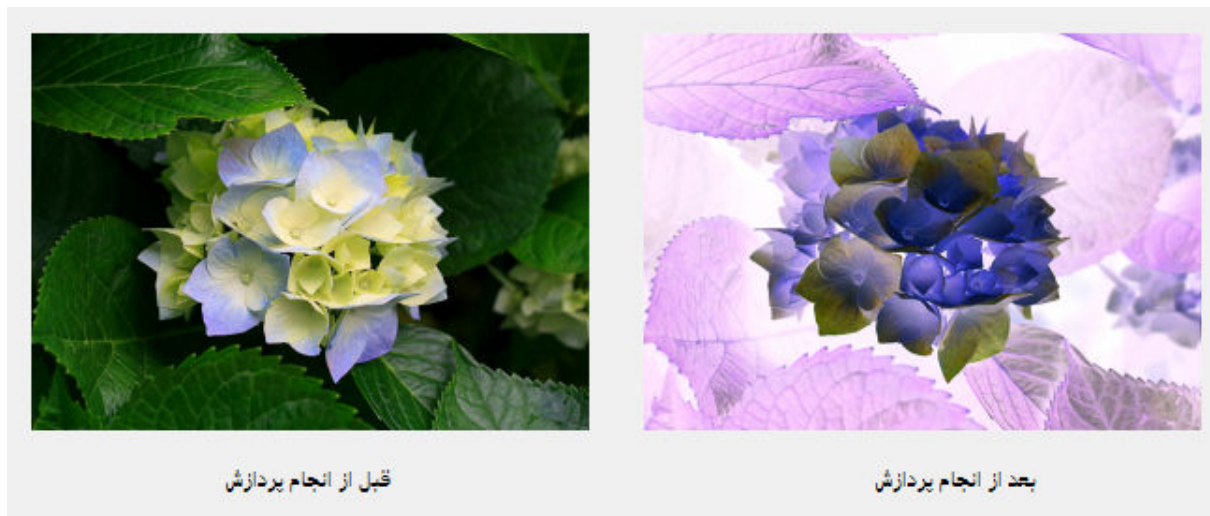
**نکته :** با استفاده از **FromArgb** می توانیم ۱۶۷۷۷۲۱۹ (۲۵۶×۲۵۶×۲۵۶) رنگ تولید کنیم !!!!

درمثال زیر ما یک عکس با اندازه دلخواه ایجاد می کنیم [2] و پیکسل های آن را به رنگ دلخواه درآورده و بر روی هارد دیسک ذخیره می کنیم.

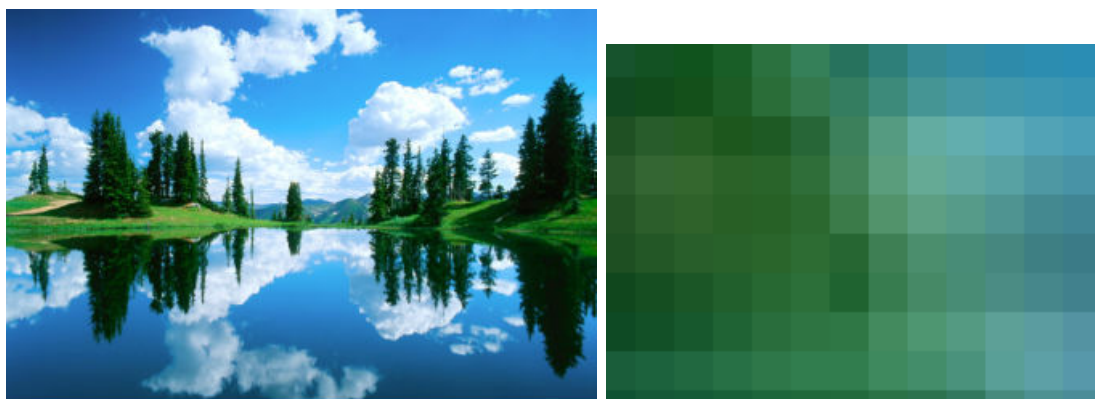
```
private void button1_Click(object sender, EventArgs e)
{
    System.Drawing.Bitmap pic = new Bitmap(600, 400);
    for (int i = 0; i < 600; i++)
    {
        for (int j = 0; j < 400; j++)
        {
            pic.SetPixel(i, j, Color.Red);
        }
    }
    pic.Save("c:\\picture.jpg",
System.Drawing.Imaging.ImageFormat.Jpeg);
}
```

همانطور که مشاهده کردید این کار به آسانی توسط کلاس های سی شارپ انجام گرفت . اگر خواستید بر روی پیکسل های عکسی کار کنید بهتر است تک تک رنگ آن پیکسل ها را گرفته و عملیاتی بر روی آن انجام دهید چون هر پیکسل از سه رنگ اصلی **RGB** ساخته شده است و هر رنگ هم در بازه بین صفر تا 256 تغییر می کند بنابراین با دستکاری این اعداد می توان پردازشی بر روی تصاویر انجام داد برای مثال در کد زیر از **rgb** هر پیکسل ۲۵۵ کم می کنیم تا عمل **Invert** بر روی عکس انجام گرفته باشد:

```
private void button1_Click(object sender, EventArgs e)
{
    System.Drawing.Bitmap pic =(Bitmap) Bitmap.FromFile("a.jpg");
    pictureBox1.Image = Bitmap.FromFile("a.jpg");
    برای نمایش عکس قبل از پردازش
    for (int i = 0; i < pic.Width; i++)
    {
        for (int j = 0; j < pic.Height; j++)
        {
            Color clr = pic.GetPixel(i, j);
            pic.SetPixel(i, j, Color.FromArgb(255-clr.R, 255-
clr.G, 255-clr.B));
        }
    }
    pictureBox2.Image = pic;
}
```



نکته: اگر خواستید با سی شارپ پردازش تصویر انجام دهید با استفاده از این کلاس به آسانی امکان پذیر است. مثال: برنامه ای بنویسید که با آن بتوان عکسی را تغییر اندازه داد؟ قبل از اینکه به این مثال پردازش ابتدا توضیحاتی در باره پردازش تصویر ارایه می دهیم. همانطور که می دانید یک عکس دیجیتالی از هزاران **pixel** تشکیل شده است که هر پیکسل هم از یک رنگ تشکیل شده است برای درک این مفهوم عکس زیر را در نظر بگیرید این عکس ۱۶۰۰ پیکسل در طول و ۱۴۰۰ پیکسل در عرض دارد در کل این عکس از ۱۹۲۰۰۰۰ پیکسل تشکیل یافته است وقتی این عکس را بزرگ کنیم شما می توانید پیکسل های آن را مشاهده فرمایید که از رنگ های متفاوتی تشکیل یافته است. بنا براین در مثال قبلی با استفاده از کلاس **bitmap** یک عکس تعریف می کنیم و پیکسل های آن را به رنگ قرمز یا هر رنگ دلخواه در می آوریم.



شما وقتی در یک برنامه نقاشی با یک قلم دارید نقاشی می کنید در واقع بر روی پیکسل ها حرکت می کنید و آن را به رنگ دلخواه در می آورید (set)



حال اگر می خواهید برنامه ای بنویسید که آن را تغییر اندازه دهد باید پیکسل های آن را تغییر دهید مثلا اگر خواستید یک عکس چهار برابر شود باید هر پیکسل آن را ۴ برابر کنید در کد زیر این فرآیند را مشاهده خواهید کرد؟

```
public static Bitmap Resize(Bitmap bmp, int Width, int Height)
{
    Bitmap Temp = (Bitmap)bmp.Clone();
    Bitmap b = new Bitmap(Width, Height, Temp.PixelFormat);

    double nX = (double)Temp.Width / (double)Width;
    double nY = (double)Temp.Height / (double)Height;

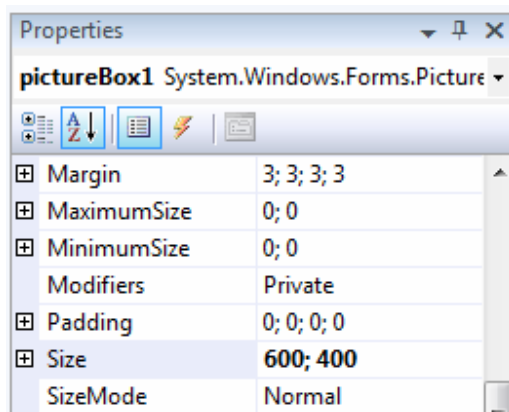
    for (int x = 0; x < b.Width; ++x)
        for (int y = 0; y < b.Height; ++y)
            b.SetPixel(x, y, Temp.GetPixel((int)(Math.Floor(x * nX)),
            (int)(Math.Floor(y * nY))));
    return b;
}
```

روش کار به این صورت است که یک تابع تعریف می کنیم و عکس را به آن تابع می فرستیم و نیز اندازه ای که قرار است به آن تغییر اندازه دهد را نیز به تابع می فرستیم و تابع عملیات تغییر اندازه را انجام می دهد برای اینکار ابتدا یک عکس جدید به اندازه ای که قرار است عکس اصلی به آن تغییر یابد تعریف می شود بعد اندازه (طول و عرض) عکس اصلی را بر اندازه ای که قرار است تغییر داده شود تقسیم می شود تا هر پیکسل به آن ضرب شود مثلا اگر قرار است یک عکس ۴ برابر شود هر پیکسل به چهار ضرب می شود. (یعنی یک پیکسل

چهار پیکسل می شود) در داخل حلقه به ترتیب بر روی پیکسل های عکس فرعی حرکت می شود و هر پیکسل آن set می شود مثلا وقتی قرار است پیکسل [5, 5] set شود در آن پیکسل رنگ های عکس اصلی در ۴ ضرب می شود البته شاید فهم این کد برای بعضی ها خیلی سخت باشد که با تمرین فهم آن آسان می شود.

مثال : می خواهیم یک برنامه نقاشی درست کنیم ؟

برای این کار یک pictureBox بر روی صفحه نمایش قرار دهید و اندازه آن را به 600\*400 تغییر دهید بعد بر روی فرم دو بار کلیک کنید تا متد رویداد آن فراخوانی شود:



هدف اینست که می خواهیم یک عکس ۶۰۰x۴۰۰ تعریف کنیم و بعد بر روی آن عملیاتی انجام دهیم و آن عکس را در pictureBox قرار دهیم اما قبل از انجام این عمل می خواهیم وقتی فرم بالا می آید رنگ عکس به کلی سفید شود. (با کد زیر)

```

Bitmap bmp = new Bitmap(600, 400);
private void Form1_Load(object sender, EventArgs e)
{
    for (int i = 0; i < 600; i++)
    {
        for (int j = 0; j < 400; j++)
        {
            bmp.SetPixel(i, j, Color.White);
        }
    }
    pictureBox1.Image = bmp;
}

```

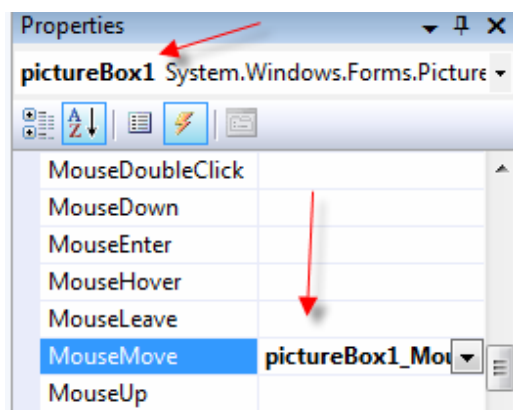
کلاسی وجود دارد که می توانیم با استفاده از آن عملیاتی بر روی عکس انجام دهیم مانند رسم خط دایره مستطیل. نام این کلاس `Graphics` است برای منتصب کردن عکس به این کلاس ابتدا یک نوع از آن تعریف می کنیم و کد زیر را بعد از کدهای قبلی اضافه می کنیم.

```

Bitmap bmp = new Bitmap(600, 400);
Graphics g;
private void Form1_Load(object sender, EventArgs e)
{
    for (int i = 0; i < 600; i++)
    {
        for (int j = 0; j < 400; j++)
        {
            bmp.SetPixel(i, j, Color.White);
        }
    }
    pictureBox1.Image = bmp;
    g = Graphics.FromImage(bmp);
}

```

حال می خواهیم با حرکت موس یک چیزی مثل مستطیل کوچک در صفحه نقاشی شود ولی برای بدست آوردن مختصات صفحه  $(X, Y)$  از رویداد `pictureBox1_MouseMove` استفاده می کنیم



و دو متغیر عمومی تعریف می کنیم که مختصات این رویداد را در آن قرار بدهیم و بعد از این متغیرها در رویداد دیگر (در صورت نیاز) برای نقاشی استفاده خواهیم کرد:

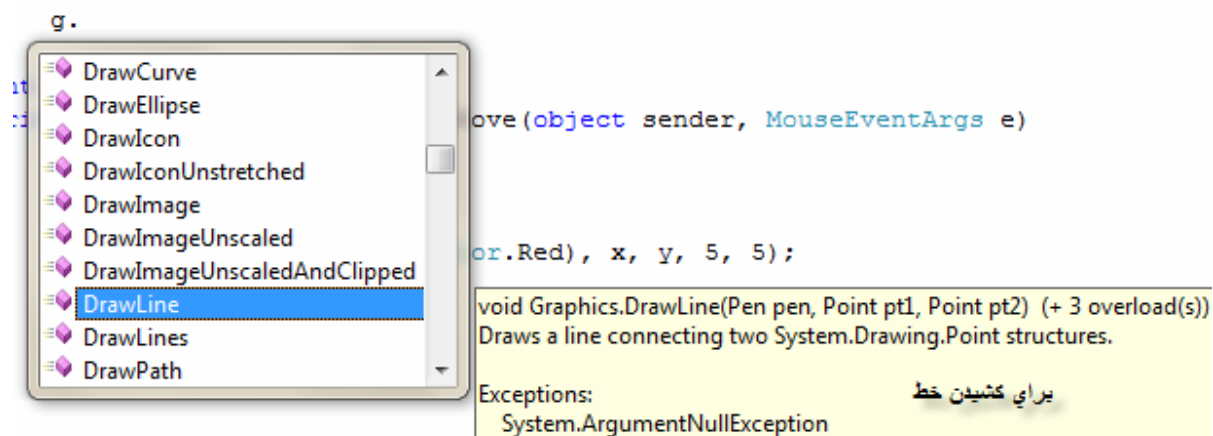
```

int x, y;
private void pictureBox1_MouseMove(object sender, MouseEventArgs e)
{
    x = e.X;
    y = e.Y;
}

```

حالا برویم سراغ کلاس Graphics

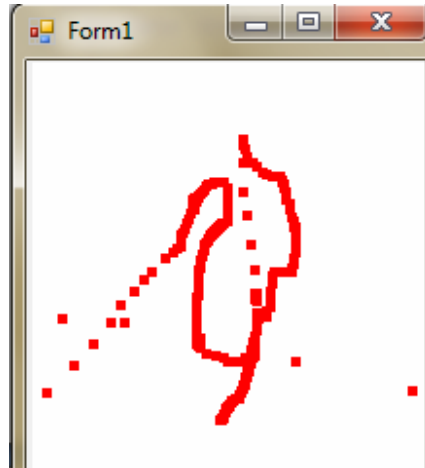
این کلاس دارای توابع مختلفی برای نقاشی است که می توانید آنها را مشاهده کنید و نیز توضیحات آن را نیز می توانید ببینید اینکه هر کدام چه وظیفه ای دارند.



ما در این مثال از متد FillRectangle استفاده میکنیم که کار این متد نقاشی یک مستطیل تو پر با اندازه دلخواه است ولی برای اینکار نیاز به یک SolidBrush است که یک کلاس از آن ایجاد می کنیم مثل کد زیر:

```
int x, y;
private void pictureBox1_MouseMove(object sender, MouseEventArgs e)
{
    x = e.X;
    y = e.Y;
    SolidBrush sl=new SolidBrush(Color.Red);
    g.FillRectangle(sl, x, y, 5, 5);
    pictureBox1.Image = bmp;
}
```

حال برنامه را اجرا کنید و با موس نقاشی کنید



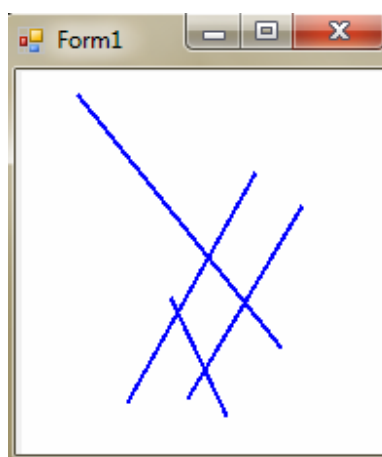
حال می خواهیم یک خط بر روی صفحه نمایش بکشیم برای اینکار نیاز به نقطه شروع و پایان است بنابراین از متد `MouseDown` و `MouseUp` یکی برای اینکه وقتی موس فشار داده شد نقطه شروع را نگه دارد و یکی برای اینکه وقتی موس برداشته شد نقطه پایان را نگه داشته و!!! خط را بکشد.

```
private void pictureBox1_MouseMove(object sender, MouseEventArgs e)
{
    x1 = e.X;
    y1 = e.Y;
}

private void pictureBox1_MouseDown(object sender, MouseEventArgs e)
{
    x2 = e.X;
    y2 = e.Y;
}
int x1, y1, x2, y2;
private void pictureBox1_MouseUp(object sender, MouseEventArgs e)
{
    Pen pn = new Pen(Color.Blue, 2);
    g.DrawLine(pn, x1, y1, x2, y2);
    pictureBox1.Image = bmp;
}
```

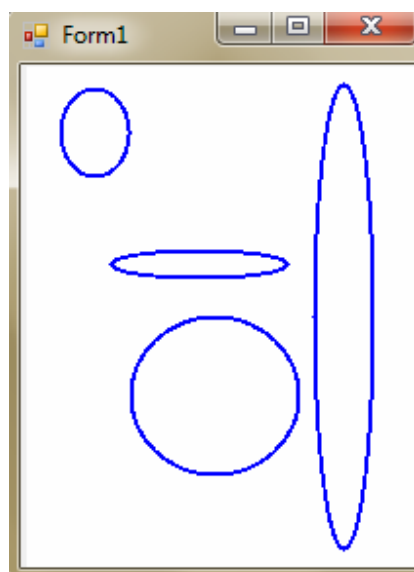


mSP soft



برای رسم بیضی

```
private void pictureBox1_MouseUp(object sender, MouseEventArgs e)
{
    Pen pn = new Pen(Color.Blue, 2);
    g.DrawEllipse(pn, x1, y1, x2-x1, y2-y1);
    pictureBox1.Image = bmp;
}
```



اگر خواستید با فشار دکمه چپ موس یا راست موس نقاشی انجام گیرد به ای شکل عمل می کنیم

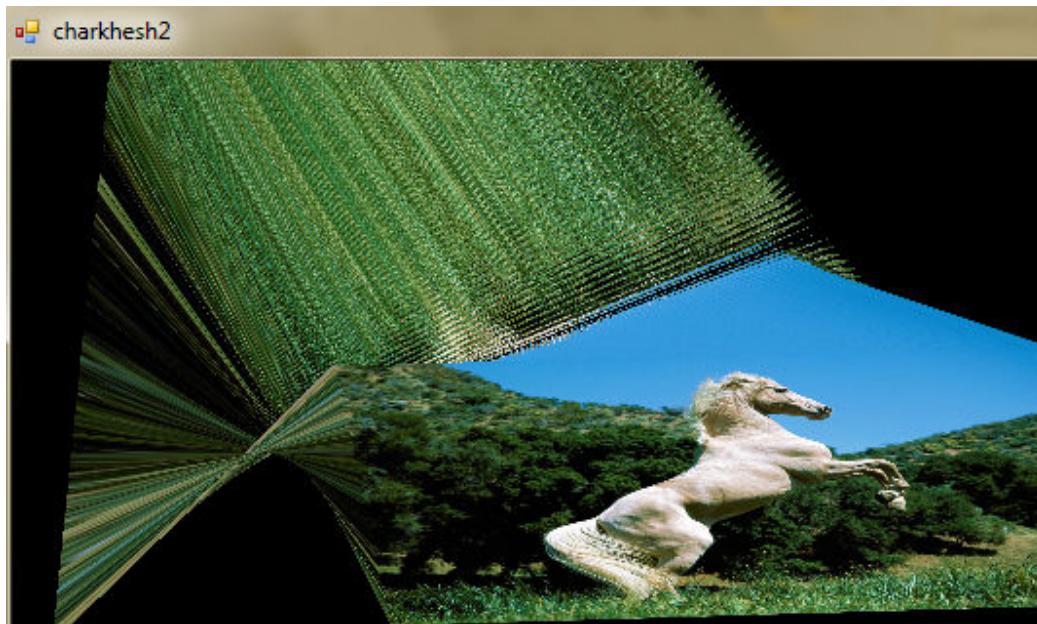
```
private void pictureBox1_MouseUp(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButton.Left)
    {
        Pen pn = new Pen(Color.Blue, 2);
        g.DrawEllipse(pn, x1, y1, x2 - x1, y2 - y1);
    }
}
```



```
        pictureBox1.Image = bmp;
    }
}
```

مثال برنامه ای که یک عکس را به صورت سه بعدی چرخش می دهد

```
public partial class charkhesh2 : Form
{
    Graphics g;
    private int a = 20, b = 300, c = 100, d = 1, ee = 50, f = 1;
    public charkhesh2()
    {
        InitializeComponent();
    }
    private void charkhesh2_Load(object sender, EventArgs e)
    {
        g = this.CreateGraphics();
    }
    private void timer1_Tick(object sender, EventArgs e)
    {
        GG();
        a++; b--; c += 3; d++; ee++; f += 2;
        if (c == 700)
        {
            timer1.Stop();
        }
    }
    private void charkhesh2_Paint(object sender, PaintEventArgs e)
    {
        GG();
    }
    public void GG()
    {
        Point[] destinationPoints = {
            new Point(a,b),
            new Point(c,d),
            new Point(ee, f),
        };
        Image image = new Bitmap(@"22.jpg");
        g.DrawImage(image, destinationPoints);
    }
    private void button1_Click(object sender, EventArgs e)
    {
        timer1.Start();
    }
    private void button2_Click(object sender, EventArgs e)
    {
        g.Clear(Color.Empty);
        timer1.Stop();
    }
}
```



مثال : برنامه ای که از صفحه نمایش عکس می گیرد:

```
private void button1_Click(object sender, EventArgs e)
{
    Bitmap _MBMP = new Bitmap(Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height);
    Graphics g = Graphics.FromImage(_MBMP);
    g.CopyFromScreen(0, 0, 0, 0, _MBMP.Size);
    pictureBox1.Image = _MBMP;
}
```

مثال : کد ترکیب دو رنگ :

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    g.SmoothingMode = SmoothingMode.HighSpeed;
    GraphicsPath gPath = new GraphicsPath();

    Rectangle r = new Rectangle(0, 0, this.Width, this.Height);
    gPath.AddRectangle(r);

    LinearGradientBrush lb = new LinearGradientBrush(r, Color.White,
Color.Black, LinearGradientMode.Horizontal);

    g.FillPath(lb, gPath);
}
```



مثال : کد مربوط به سیاه و سفید کردن عکس :

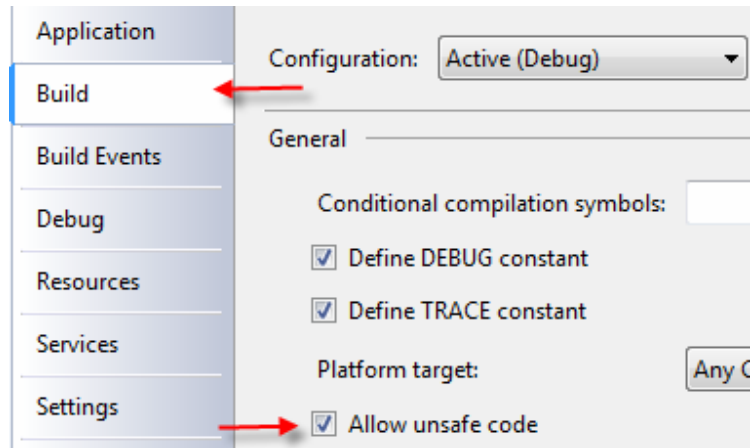
```

public void greyscale(Bitmap bmp, PictureBox picBox)
{
    BitmapData data = bmp.LockBits(new Rectangle(0, 0, bmp.Width,
bmp.Height),
    ImageLockMode.ReadWrite, PixelFormat.Format24bppRgb);
    unsafe
    {
        byte* imgPtr = (byte*)(data.Scan0);
        byte red, green, blue;
        for (int i = 0; i < data.Height; i++)
        {
            for (int j = 0; j < data.Width; j++)
            {
                blue = imgPtr[0];
                green = imgPtr[1];
                red = imgPtr[2];

                imgPtr[0] = imgPtr[1] = imgPtr[2] =
                    (byte)(.299 * red
                        + .587 * green
                        + .114 * blue);
                imgPtr += 3;
            }
            imgPtr += data.Stride - data.Width * 3;
        }
    }
    bmp.UnlockBits(data);
    picBox.Image = bmp;
}

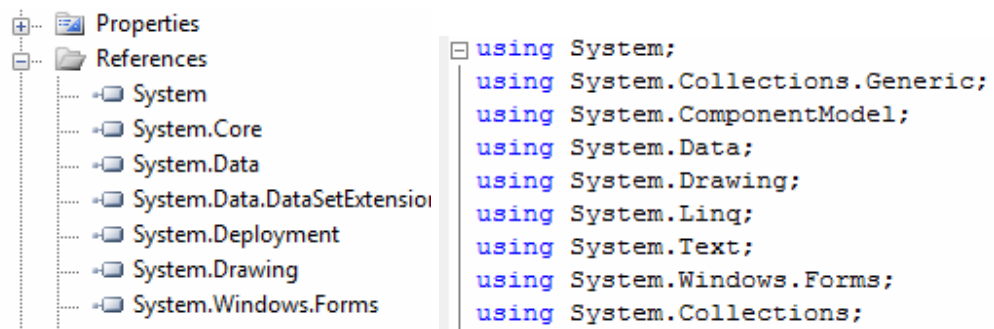
```

چون کد بالا به صورت کدهای نا امن نوشته شده باید از قسمت Properties تیک Allow unsafe code را فعال کنید تا برنامه کامپایل شود.



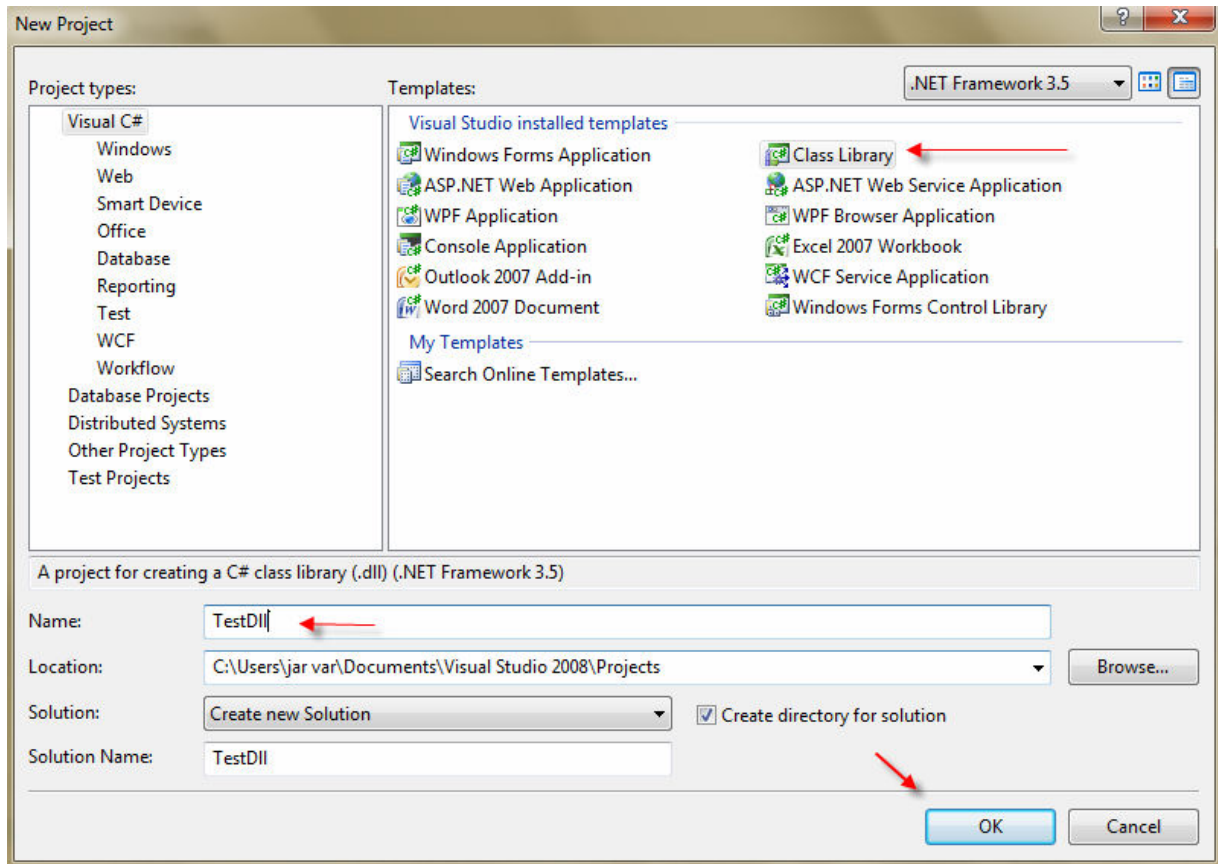
## ۱۴-۱DLL

همه هدرهایی که در محیط برنامه نویسی در بالا مشاهده می کنید در قالب dll ذخیره شده اند

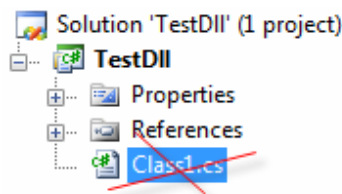


و در برنامه مورد استفاده قرار می گیرند که در اینجا می خواهیم به نحوه ساختن آنها و استفاده در برنامه پردازیم. وقتی ما dll می سازیم در همه برنامه ها می توانیم از آنها استفاده کنیم و حتی در اختیار دیگران برای استفاده قرار دهیم. در ساختن dll ها می توانیم از کلاس های متفاوتی داخل آن استفاده کنیم حتی کلاس فرم. همانطور که ملاحظه کردید dll (System.Collection) دارای کلاسهای متفاوتی بود از جمله کلاس ArrayList و Stack. اگر به قسمت C:\Windows\System32 ویندوز مراجعه کنید با انواع dll ها برخورد خواهید کرد که سیستم عامل از آنها استفاده می کند.

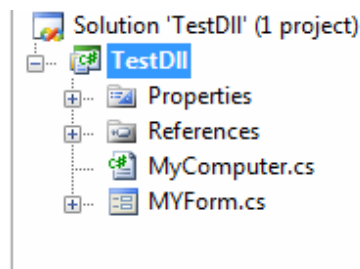
برای ساختن dll ها کافیست یک برنامه جدید از نوع ClassLibrary ایجاد کنید



حال class1 را نیز حذف کنید تا خودمان یک کلاس با نام دلخواه اضافه کنیم



و یک کلاس با نام MyComputer و یک فرم با نام MYForm به برنامه اضافه کنید



در داخل کلاس خود یک چیز دلخواه بنویسید مثلا کد زیر که یک تابع است و یک استرینگ بر می گرداند

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

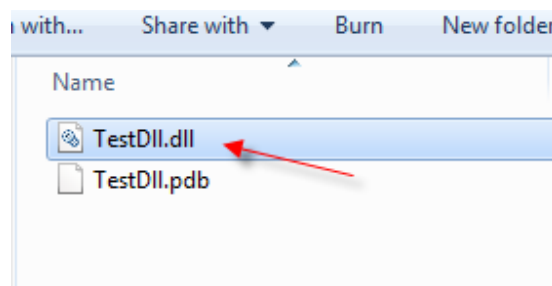
namespace TestDll
{
    public class MyComputer
    {
        public string mas ()
        {
            return "my name is ...!!!";
        }
    }
}
```

فقط به این نکته توجه داشته باشید که حتما `public` باشد هم کلاس و هم تابع !!!!

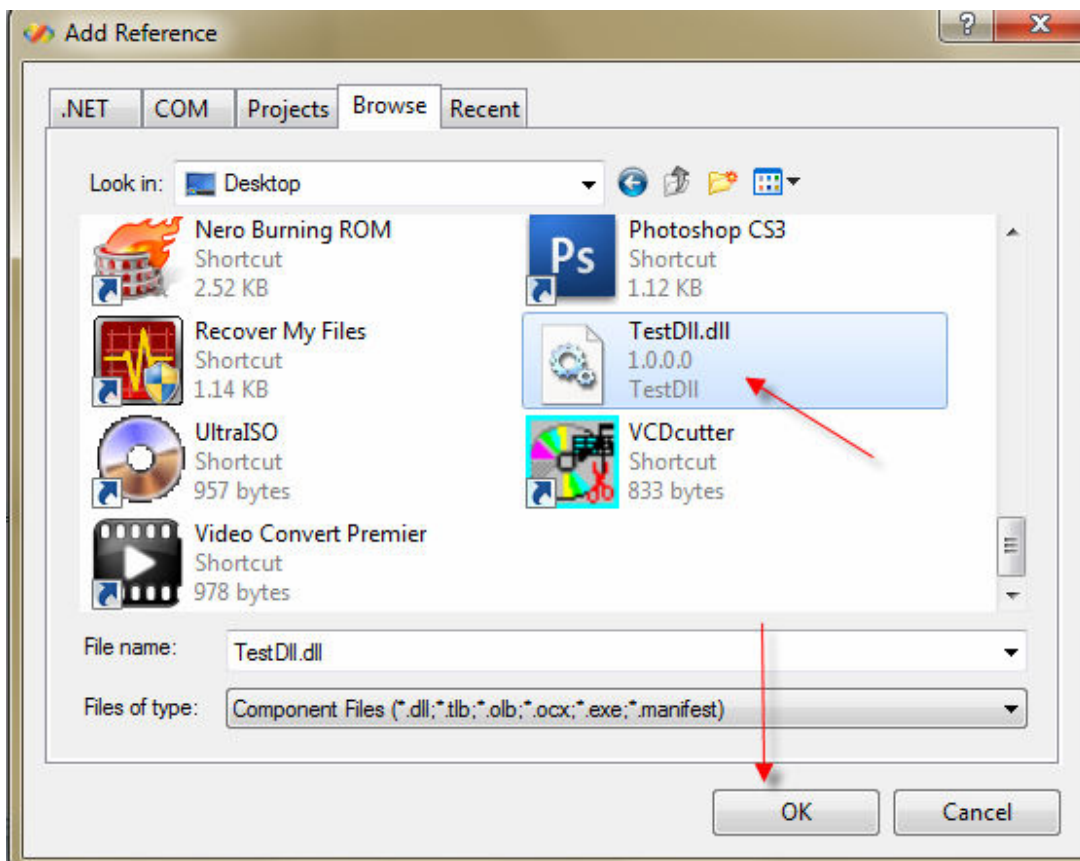
بعد بر روی فرم یک دکمه و یک تکست باکس قرار داده و کد زیر را برای دکمه بنویسید :

```
private void button1_Click(object sender, EventArgs e)
{
    MyComputer mc = new MyComputer();
    textBox1.Text = mc.mas();
}
```

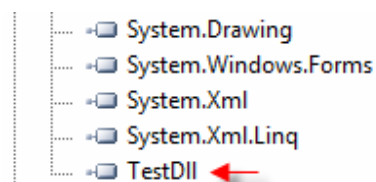
بر روی نام برنامه کلیک راست کرده و `Build` را بزنید تا `dll` شما ساخته شود حال از برنامه خارج شود و به مسیری که برنامه در آنجا ایجاد شده بود بروید و فایل `DLI` را بردارید و در جایی از هارد دیسک خود ذخیره کنید.



حال می خواهیم ببینیم چگونه می توانیم از این dll در دیگر برنامه ها استفاده کنیم. برای اینکار پروژه جدید ایجاد کنید و و از قسمت References کلیک راست کرده و Add references را کلیک کنید و بعد فایل dll خود را که در جایی ذخیره کردید انتخاب کنید



تا اضافه شود



بعد نام هدر مورد نظر را به برنامه using کنید :

```
using TestDll;
```

حال می توانید از کلاس های داخلی این هدر استفاده کنید مثال:





در این قسمت به بحث جالب ترد ها می پردازیم اینکه مفهوم ترد ها چیست و چگونه می توانیم آن ها را کنترل کنیم . تا حالا برای شما پیش آمده که چندین برنامه را با هم اجرا کنید مثلا وقتی در حال بازی کردن یا تایپ مقاله هستید همزمان به موسیقی هم از طریق مدیا پلیر گوش می دهید ولی CPU در هر لحظه می تواند یک محاسبه را انجام بدهد پس اینکار چگونه انجام می گیرد ؟ در واقع همینطور است و CPU در هر لحظه یک دستور را اجرا می کند ولی این اولویت بندی برنامه ها است که این ها را از هم جدا می کند چون سرعت اجرای CPU بسیار زیاد است ما قادر به درک جابجایی برنامه نیستیم . البته وظیفه این کار بر عهده سیستم عامل است که چگونه بتواند وقت CPU را بین برنامه ها تقسیم کند.

در این قسمت وبا استفاده از Threading می خواهیم ببینیم چگونه می توانیم همزمان چند دستور را با هم اجرا کنیم اگر ترد ها نبود اینکار ممکن نبود چون همانطور که می دانید وقتی برنامه ای می نویسید این برنامه ردیفی اجرا می شود مگر در لحظاتی که پرش داشته باشیم مانند پرش به تابع و برگشت به مکانی که از آنجا پرش انجام گرفته و در این صورت هم همزمانی انجام نگرفته حالا با مثالی با مفهوم ترد ها بیشتر آشنا می شویم.

مثال :

برنامه ای بنویسید که همزمان عمل کپی کردن دو فایل را انجام بدهد مگر اینکه ما اولیت را به یکی بدهیم ؟

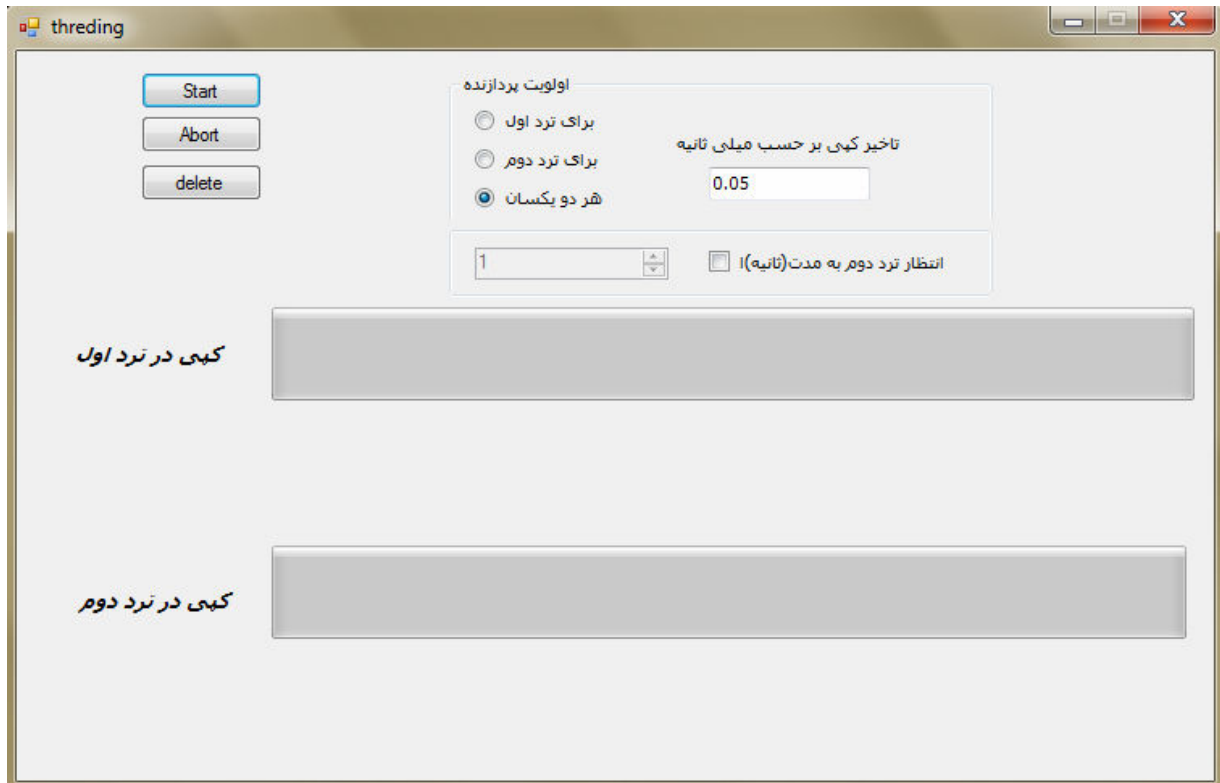
برای حل این مثال ابتدا دو هدر زیر را به برنامه اضافه کنید:

```
using System.IO;
using System.Threading;
```

در هدر IO از دو کلاس BinaryReader و BinaryWriter برای کپی کردن فایل استفاده می کنیم .

برای انجام اینکار دو progressBar و سه radioButton و دو groupBox بر روی فرم قرار دهید یکی از کار groupBox اینست که اگر radioButton ها داخل این کنترل قرار گیرند در هنگام اجر فقط یکی از radioButton ها اجرا می شود .

و بقیه کنترل ها را مطابق شکل تنظیم کنید



روش کار به این صورت است که دو شی از کلاس ترد تعریف می کنیم کار این دو کلاس اینست که تابعی را اجرا کنند پس دو تابع هم شکل تعریف می کنیم که عمل کپی برداری را انجام بدهند اما چون کلاس ترد به طور مستقیم نمی تواند تابع را اجرا کند از کلاس دیگری برای همین منظور استفاده می کنیم با نام `ThreadStart` و بعد این کلاس را به کلاس ترد منتصب می کنیم .

حال کد های زیر را برای دکمه اول `start` می نویسیم

```
Thread t1, t2;
double speed = 0.05;
private void button1_Click(object sender, EventArgs e)
{
    ThreadStart ts2 = new ThreadStart(Copy);
    ThreadStart ts3 = new ThreadStart(Copy2);
    t1 = new Thread(ts2);
    t2 = new Thread(ts3);
    if (radioButton1.Checked)
    {
        t2.Priority = ThreadPriority.Lowest;
        t1.Priority = ThreadPriority.Highest;
    }
    if (radioButton2.Checked)
    {
```

```

        t1.Priority = ThreadPriority.Lowest;
        t2.Priority = ThreadPriority.Highest;
    }
    else
    {
        t2.Priority = ThreadPriority.Highest;
        t1.Priority = ThreadPriority.Highest;
    }
    speed = Convert.ToDouble(textBox1.Text);
    t1.Start();
    if (checkBox1.Checked)
    {
        t1.Join((int)numericUpDown1.Value * 1000);
    }
    else
    {
    }
    t2.Start();
}

```

کد های دیگر برای تنظیم سرعت کپی و اولویت به ترد ها است . و نیز مدت Start ترد برای شروع کار ترد استفاده می شود چون ترد در اول به طور اتوماتیک اجرا نمی شود.

از جمله کار های دیگر ترد ایجاد وقفه است یعنی در یک خطی ما وقفه ای ایجاد می کنیم وقتی برنامه به آن خط رسید به مدتی که تعیین کردیم ایست می کند و بعد اجرا می شود. از این روش در تابعهای کپی استفاده کردیم تا وقفه ای به برنامه ها به مدت چند ثانیه بدهیم تا نتیجه کار را مشاهده کنیم .

حال کدهای دو تابع را که یکسان است می نویسیم . کار این تابعها اینست که فایلهایی را که مشخص کردیم را به مقصد مشخص شده کپی می کند برای انجام آزمایش دو فایل با نامهای مشخص شده در تابع رادر کنار فایل اجرایی قرار دهید .

```

public void Copy()
{
    int NumRead;
    long FileLength;
    FileStream From = new FileStream("1.iso", FileMode.Open);
    FileStream To = new FileStream("3.iso",
    FileMode.CreateNew);
    byte[] buffer = new byte[1024];
    FileLength = From.Length;
    progressBar1.Minimum = 0;
    progressBar1.Maximum = (int)FileLength;
    while (FileLength > 0)
    {
        System.IO.BinaryReader Reader = new
        System.IO.BinaryReader(From);

```

```

        NumRead = Reader.Read(buffer, 0, 1024);
        FileLength = FileLength - NumRead;

        System.IO.BinaryWriter Writer = new
        System.IO.BinaryWriter(To);
        Writer.Write(buffer, 0, NumRead);
        progressBar1.Value = progressBar1.Value + NumRead;
        Writer.Flush();
        Thread.Sleep(TimeSpan.FromMilliseconds(spead));
    }
    From.Close();
    To.Close();
    if (progressBar1.Value > 99)
    {
        progressBar1.Value = 0;
        MessageBox.Show("Copy Finished successfully");
    }
}
////////////////////////////////////
public void Copy2()
{
    int NumRead;
    long FileLength;
    FileStream From = new FileStream("2.iso", FileMode.Open);
    FileStream To = new FileStream("4.iso",
    FileMode.CreateNew);
    byte[] buffer = new byte[1024];
    FileLength = From.Length;
    progressBar2.Minimum = 0;
    progressBar2.Maximum = (int)FileLength;
    while (FileLength > 0)
    {
        System.IO.BinaryReader Reader = new
        System.IO.BinaryReader(From);

        NumRead = Reader.Read(buffer, 0, 1024);
        FileLength = FileLength - NumRead;

        System.IO.BinaryWriter Writer = new
        System.IO.BinaryWriter(To);
        Writer.Write(buffer, 0, NumRead);
        progressBar2.Value = progressBar2.Value + NumRead;
        Writer.Flush();
        Thread.Sleep(TimeSpan.FromMilliseconds(spead));
    }
    From.Close();
    To.Close();
    if (progressBar2.Value > 99)
    {
        progressBar2.Value = 0;
        MessageBox.Show("Copy Finished successfully");
    }
}
}

```

روش کار تابع به این صورت است که در هر لحظه یک کیلو بایت از فایل را خوانده و آن را کپی می کند اینکار تا پایان فایل انجام می گیرد. می توانید هر چند کیلو بایت که خواستید عمل کپی را انجام بدهید حتی می توانید یک جا نیز کپی کنید ولی در این کد برای مشاهده نتیجه و استفاده کمتر از فضای رم کد اینگونه نوشته شده است . با استفاده از

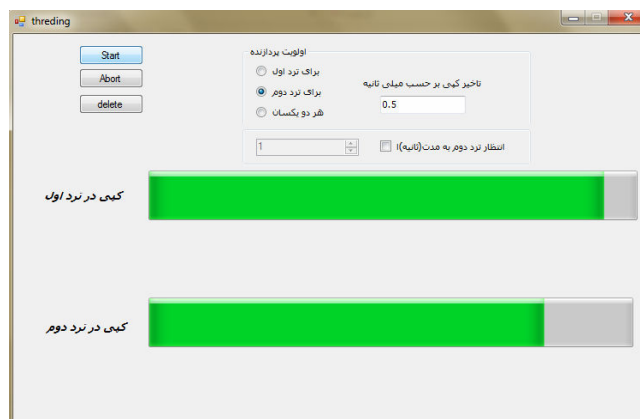
```
Thread.Sleep(TimeSpan.FromMilliseconds(spead));
```

تاخیری در برنامه به مدت چند میلی ثانیه ایجاد کردیم تا برنامه با ماکسیم سرعت CPU اجرا نشود.

اما روش کار progressBar به این صورت است که یک ماکسیمم برای این کنترل در نظر می گیریم مثلا ۳۰۰۰ و progressBar خودش به نوعی این مقدار را با پر شدن تدریجی متناسب می کند مثلا اگر از صفر شروع کنیم تا سه هزار اگر با اضافه کردن به عدد صد برسیم progressBar یک خانه را رنگ می کند.













نکته : وقتی با ترد ها کار می کنید و از کنترل هایی مانند progressBar استفاده می کنید حتما برنامه را با

Ctrl+F5 اجرا کنید تا برنامه error ندهد چون این کنترل ها در داخل خودشان نیز از ترد استفاده می کنند.



## ۱-۱۶ کار با فایلها :

تا حالا با فایل های زیادی کار کردید مثلا فایل های متنی مانند (word, txt, pdf) یا فایل های مدیا مانند (mp3, divx, mpeg) یا فایل های تصویری مانند (bitmap, jpg, png, gif) .

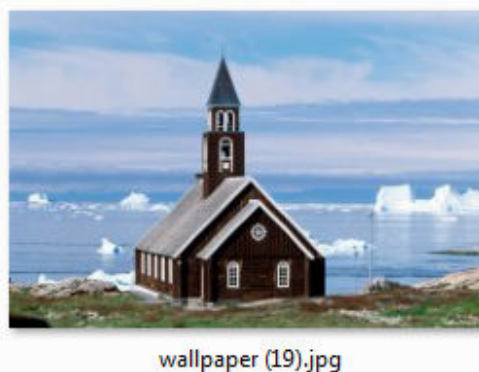
Name	Type	Size
 clip (185).mpg	MPG File	36,015 KB
 f.mp3	MP3 File	6,224 KB
 image54.jpg	JPEG image	39 KB
 LAST C.pdf	Adobe Acro...	708 KB
 Nastaliq.ttf	TrueType fo...	184 KB
 new.rar	WinRAR arc...	1,036 KB
 report.doc	Microsoft O...	778 KB
 report.docx	Microsoft O...	758 KB
 sound6.wav	Wave Sound	5,873 KB
 test.txt	Text Docum...	1 KB
 Untitled.bmp	Bitmap image	3,001 KB
 Welcome to QuickTime.mov	QuickTime ...	1,260 KB

ولی آیا تا به حال به این فکر کردید که این فایلها چگونه ساخته شده اند و چگونه نمایش داده می شوند. آیا تا به حال کلمه فشرده سازی به گوشتان خورده حتما شنیدید که می گویند mp3 فشرده سازی شده ولی شاید ندونید مفهوم فشرده سازی یعنی چه؟ در این بخش می خواهیم به این بحث ها در سطح یادگیری بپردازیم .

همه فایلها حتی فایل اجرایی exe به وسیله کدهای باینری ساخته و ذخیره می شوند اما آن چیزی که اینها را از هم متمایز می کند نحوه ذخیره سازی آنها است برای درک بهترین مطلب توجه شما را به دو عکس زیر جلب می کنم که یکی با فرمت jpg ذخیره شده و دیگری با فرمت bmp.



**7.62 MB**



**521 KB**

در ظاهر هر دو عکس یکی هستند ولی اگر به ظرفیت آنها توجه کنید می بینید که ظرفیت دومی تقریباً ۱۴ برابر اولی است!!! دلیل این تفاوت بر می گردد به نحوه ذخیره سازی آنها. در عکس با فرمت bmp هیچ گونه فشرده سازی انجام نگرفته است ولی در jpg فشرده سازی انجام گرفته است. در bmp عکس پیکسل به پیکسل ذخیره شده است یعنی هر پیکسل یک فضایی در حافظه اشغال کرده است ولی در jpg قضیه فرق می کند برای فهم این مساله توجه شما را به هر دو عکس جلب می کنم اگر به عکس ها نگاه کنید می فهمید که چندین پیکسل در فضای بالای عکس (آسمان) به رنگ آبی است مثلاً از پیکسل [0,0] تا پیکسل [100,100] خوب حالا چه ضرورتی دارد که همه پیکسل ها را تا نقطه [100,100] یکی یکی ذخیره کنیم با اینکه می دانیم همه آبی است!! مثلاً اگر هر پیکسل ۲۴ بیت فضا اشغال کند در کل برای ذخیره سازی این ۱۰۰۰۰ پیکسل به ظرفیتی حدود ۳۰ کیلو بایت نیاز داریم ولی در صورتی که در ذخیره سازی بگوییم که از خانه صفر تا [100,100] پیکسل های ما آبی است به ظرفیتی حدود ۵۰ بایت نیاز داریم!!!! (البته در jpg روش ذخیره سازی به این صورت نیست) این مثال را گفتم که با مفهوم فشرده سازی بهتر آشنا شوید. خوب حالا یکی پیدا می شود که می گوید من به این روش ذخیره نمی کنم بلکه با روش دیگری ذخیره می کنم که فضای کمتری اشغال می کند به این شکل است که فرمت های مختلفی شکل می گیرد. توجه: وقتی برنامه را با فرمتی ذخیره می کنیم باید برنامه ای هم بنویسیم که آن فرمت را برای ما بخواند چون دیگران نمی دانند ما چگونه ذخیره کردیم تا آن را باز کنند. مثلاً اگر ما بدانیم که فایل jpg چگونه ذخیره شده است می توانیم برنامه ای بنویسیم که آن را نمایش دهد.

مثال: یک فایل با پسوند (mer) ساخته و نام خود را در آن ذخیره کنید بعد آن را باز کرده و در یک تکست باکس نمایش دهید؟

برای انجام اینکار از دو کلاس `System.IO.BinaryReader` و `System.IO.BinaryWriter` یکی برای خواندن از فایل و یکی برای نوشتن و ایجاد فایل استفاده می شود این دو کلاس در سازنده های خود نیاز به یک کلاس از `System.IO.FileStream` (برای دادن مشخصات فایل) دارند. روش کار این دو کلاس به این صورت است که مشخصات فایل را چه برای نوشتن و چه برای خواندن در یک آرایه ای از بایت ها ویا کاراکترها قرار می دهد حالا طول آرایه بستگی به خودمان دارد که چه چند بایت چند بایت بخوانیم چون



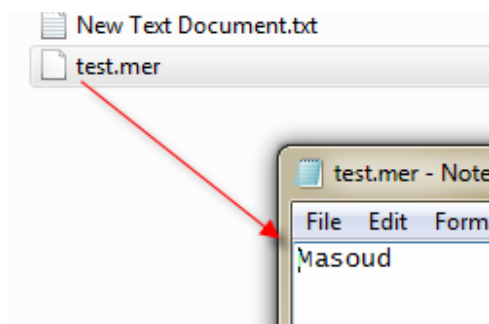
بعضی وقت ها نیاز است که تکه تکه بخوانیم و عملیاتی روی آنها انجام بدهیم ویا برعکس. حالا اگر بخواهیم یک رشته مثل نام را در یک فایل ذخیره کنیم باید ابتدا آن را به آرایه ای از کاراکتر ها در بیاوریم و بعد ذخیره کنیم برای تبدیل یک استرینگ به آرایه ای از کاراکتر ها از متد `ToCharArray` استفاده می شود.

نکته !! : هر کاراکتر انگلیسی و بعضی از کاراکتر ها خاص یک بایت فضا اشغال می کنند بنابراین می توانیم با یک تبدیل ضمنی آن را در نوع بایت ذخیره کنیم .

در زیر کد کامل ذخیره در یک فایل با نام و پسوند دلخواه برای دکمه ذخیره نوشته شده است .

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Create);
    System.IO.BinaryWriter _BiWr = new System.IO.BinaryWriter(_FM);
    string Str="Masoud";
    char[] _CH = Str.ToCharArray();
    _BiWr.Write(_CH);
    _BiWr.Close();
    MessageBox.Show("Finish");
}
```

حال فایل شما ایجاد شد و اگر خواستید می توانید در یک notepad باز کنید و ببینید



حال اگر خواستید تبدیل به یک آرایه از بایت کنید و بعد ذخیره کنید که هیچ فرقی ندارد مثل کد زیر:

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Create);
    System.IO.BinaryWriter _BiWr = new System.IO.BinaryWriter(_FM);
    string Str="Masoud";
    char[] _CH = Str.ToCharArray();
    byte[] _BY = new byte[_CH.Length];
    for (int i = 0; i < _CH.Length; i++)
```

```

{
    _BY[i] = (byte)_CH[i];
}
_BiWr.Write(_BY);
_BiWr.Close();
MessageBox.Show("Finish");
}

```

حال برنامه ای می نویسیم که این فایل را برای ما باز کند و نمایش دهد

```

private void button2_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Open);
    System.IO.BinaryReader _BiRe = new System.IO.BinaryReader(_FM);
    byte[] _BY = new byte[1];
    int j;
    string str="";
    while ((j = _BiRe.Read(_BY,0,1)) > 0)
    {
        str += _BY[0].ToString();
    }
    _BiRe.Close();
    textBox1.Text = str;
}

```

ولی چون تبدیل ضمنی انجام ندادیم کاراکتر های ما به صورت عدد (باینری) نمایش داده می شوند :

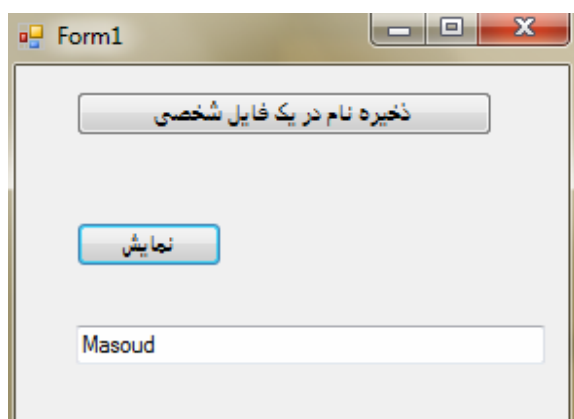


نکته : همانطور که گفتیم همه چیز در کامپیوتر به صورت صفر و یک (باینری) ذخیره می شود مثلا معادل کاراکتر (a=97) است که وقتی عدد 97 را به طور متوالی تقسیم بر 2 کنید به معادل باینری آن نیز دست خواهید یافت (01100001) و کاراکتر b=98 است به همین ترتیب اگر اضافه کنید معادل آنها را به دست خواهید آورد.

نکته : اگر از هر کاراکتر ۳۲ تا کم کنید به حروف بزرگ تبدیل می شود .

حالا بایک تبدیل ضمنی کدمان را به این شکل عوض می کنیم :

```
private void button2_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Open);
    System.IO.BinaryReader _BiRe = new System.IO.BinaryReader(_FM);
    byte[] _BY = new byte[1];
    int j;
    string str="";
    while ((j = _BiRe.Read(_BY,0,1)) > 0)
    {
        str +=((char) _BY[0]).ToString();
    }
    _BiRe.Close();
    textBox1.Text = str;
}
```



کاری که تا حالا انجام دادیم یک کار ساده ای بود و هرکس به سادگی می تواند فایل ما باز کند و ببیند . حالا می خواهیم یک روش ساده ای به کار بریم (برای یادگیری) که دیگر هر کس به سادگی نتواند محتویات فایل ما را ببیند .

همانطور که در بالا دیدید معادل حروف انگلیسی کاراکتر  $a=01100001$  است و اگر به دقت نگاه کنید می فهمید که بیت آخر همه ۲۶ کاراکتر انگلیسی صفر است بنابراین اگر بر فرض مثال آن بیت را یک کنیم دیگر این کاراکتر ها تبدیل به کاراکتر های دیگر می شوند دراین صورت دیگر وقتی فایل باز شود چیزی دیده

نمی شود که ما ذخیره کردیم. و وقتی خودمان می خواهیم فایل را باز کنیم آن بیت آخر را دوباره صفر می کنیم تا به کاراکتر اصلی برسیم .

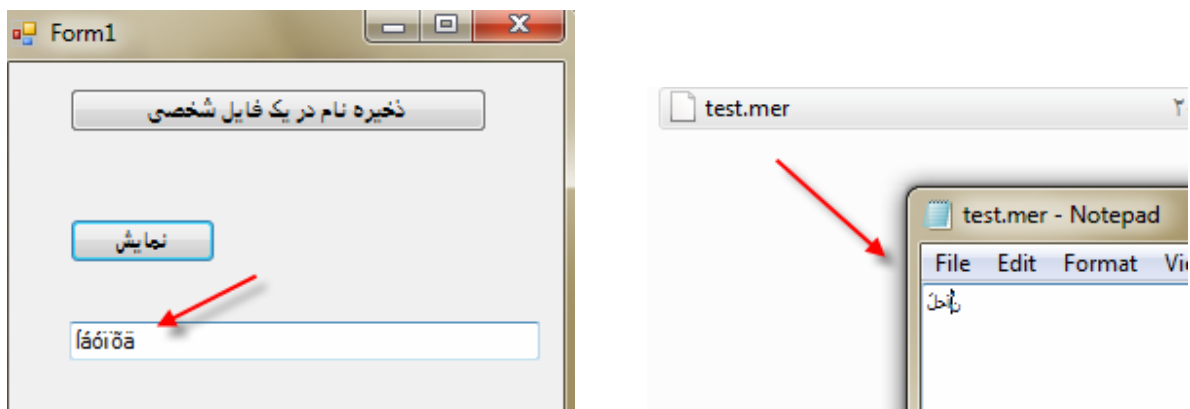
برای یک کردن بیت آخر همه آنها را با (10000000=128) OR ( | ) می کنیم و درموقع خواندن آن ها را با (01111111=127) And (&) می کنیم تا مقدار اولیه به دست بیاید. اگر با حاصل and , or آشنایی ندارید در جداول زیر حاصل آنها را آورده ایم .

A	B	OR	And
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

حال کد قبلی را به مقابل تغییر دهید :

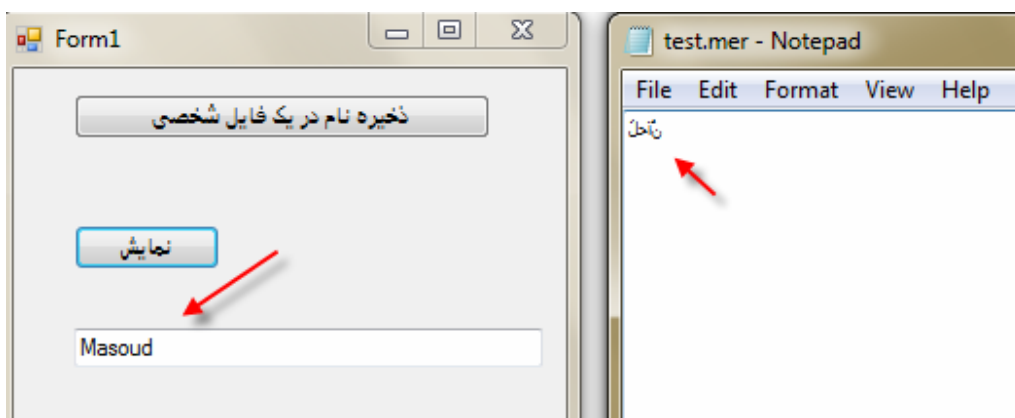
```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Create);
    System.IO.BinaryWriter _BiWr = new System.IO.BinaryWriter(_FM);
    string Str="Masoud";
    char[] _CH = Str.ToCharArray();
    byte[] _BY = new byte[_CH.Length];
    for (int i = 0; i < _CH.Length; i++)
    {
        _BY[i] = (byte)(_CH[i] | 128);
    }
    _BiWr.Write(_BY);
    _BiWr.Close();
    MessageBox.Show("Finish");
}
```

دیگر اگر کسی فایل را باز کند دیگر هیچ چیز نمی فهمد



حال کد نمایش را به این شکل تغییر دهید :

```
private void button2_Click(object sender, EventArgs e)
{
    System.IO.FileStream _FM = new System.IO.FileStream("test.mer",
System.IO.FileMode.Open);
    System.IO.BinaryReader _BiRe = new System.IO.BinaryReader(_FM);
    byte[] _BY = new byte[1];
    int j;
    string str="";
    while ((j = _BiRe.Read(_BY,0,1)) > 0)
    {
        str +=((char) (_BY[0] & 127)).ToString();
    }
    _BiRe.Close();
    textBox1.Text = str;
}
}
```

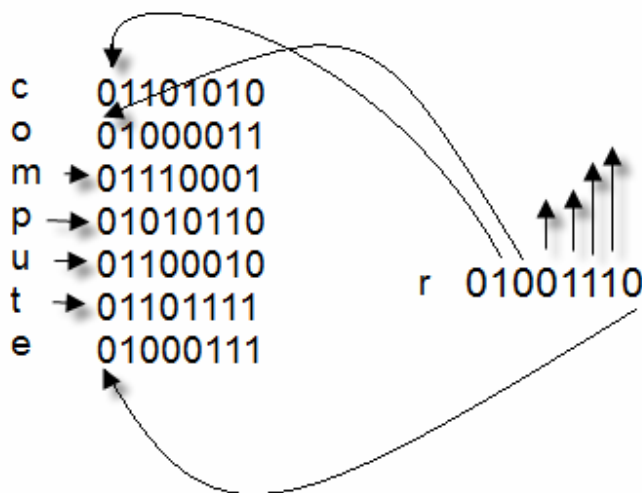


جالبه !!!!!!!

چیزی که در مثال بالا حل کردیم فشرده سازی نبود بلکه نوعی رمز کردن فایل بود اگر خواستید خودتان می توانید الگوریتم هایی را پیشنهاد بدید هم برای فشرده سازی و هم برای رمزنگاری. مثلا اگر در مثال بالا

خواستید فشرده سازی انجام بدید باید به ای صورت عمل کنید: (فقط توضیح می دهم کد نویسی به عنوان تمرین برعهده خودتان است )

چون بیت آخر همه کاراکتر های انگلیسی صفر است می توانیم در هر هشت کاراکتر یکی را در آخرین بیت ذخیره کنید مثلا اگر رشته شما به این صورت است computer می توانید کاراکتر r را در هفت بیت آخر هر کاراکتر compute ذخیره کنید و کاراکتر r را حذف کنید مانند شکل ۲۴-۱



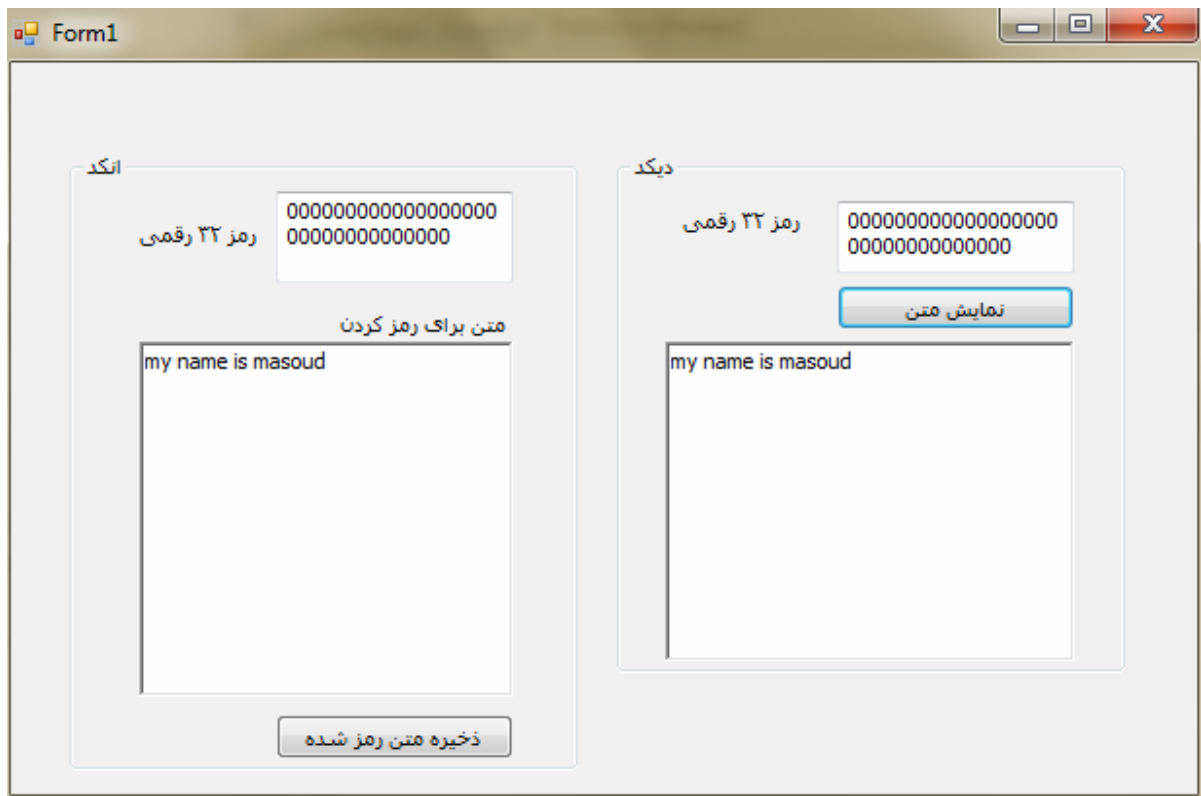
۲۴-۱

در سی شارپ چندین کلاس آماده وجود دارد هم برای فشرده سازی و هم برای رمز نگاری :

برای مثال برای رمز کردن متن خود می توانید با کلاس Rijndael این کار را با نهایت قدرت انجام بدهید این الگوریتم در سال ۹۹ اختراع شد و تاکنون هیچ الگوریتمی بهتر از Rijndael اختراع نشده است نحوه عملکرد این الگوریتم به این صورت است که ما یک رمز ۲۵۶ بیتی به این الگوریتم می دهیم و الگوریتم بر اساس رمز ما متن وارده را کد می کند و برای دیکد کردن هم به این رمز نیاز داریم

مثال : یک متنی را با استفاده از الگوریتم Rijndael انکد و دیکد کنید :

شکل برنامه را به صورت شکل ۲۵-۱ ایجاد کنید



همانطور که در شکل می بینید این برنامه یک رمز ۳۲ رقمی دریافت (۳۲ بیتی)(۲۵۶بیتی) میکند و با توجه به آن رمز عملیات کدینگ را برای متن وارد در پایین انجام میدهد و در هنگام دیکد کردن هم با توجه به آن رمز عملیات دیکرد را انجام میدهند در اینجا رمز اولیه را برای همه صفر در نظر گرفتیم که می توانید آن را تغییر دهید ولی باید به این نکته توجه کنید که در هنگام دیکد هم همان رمز را وارد کنید؟

حال کد برنامه را برای دکمه اول بنویسید :

```
using System.Security.Cryptography;
using System.IO;

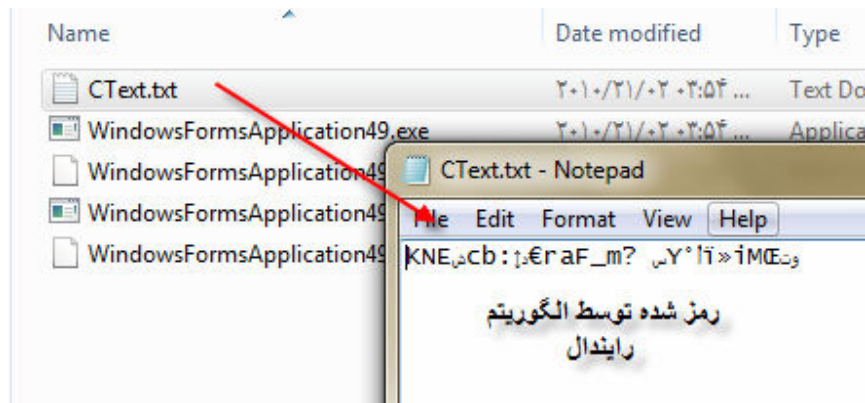
private void button1_Click(object sender, EventArgs e)
{
    Rijndael RJ = Rijndael.Create();
    string strRamz = textBox1.Text; رمز برای کد کردن متن
    string StrMatn = richTextBox1.Text; متن اصلی
    string FileName = "CText.txt"; نام فایل برای ذخیره متن کدشده
    char[] _ch = textBox1.Text.ToCharArray(); تبدیل رمز به آرایه بایت
    byte[] _by = new byte[32];
    for (int i = 0; i < 32; i++)
    {
        _by[i] = (byte)_ch[i];
    }
}
```

```

byte[] _byIV = new byte[16];for vector(16 byte need)?
for (int i = 0; i < 16; i++)
{
    _byIV[i] = 0;
}
RJ.Key = _by;
RJ.IV = _byIV;
EncryptTextToFile(StrMatn, FileName, RJ.Key, RJ.IV);
MessageBox.Show("finish");
}

```

وظیفه این رویداد دکمه ذخیره متن رمز شده بر روی یک فایل با نام `CText.txt` درکنار فایل اجرایی است .



کار اصلی آنکد کردن را تابعی انجام میدهند که با نام `EncryptTextToFile` فراخوانی می شود و دارای کد زیر است [1]:

```

public void EncryptTextToFile(String Data, String FileName, byte[]
Key, byte[] IV)
{
    try
    {
        // Create or open the specified file.
        FileStream fStream = File.Open(FileName,
        FileMode.OpenOrCreate);

        // Create a new Rijndael object.
        Rijndael RijndaelAlg = Rijndael.Create();

        // Create a CryptoStream using the FileStream
        // and the passed key and initialization vector (IV).
        CryptoStream cStream = new CryptoStream(fStream,
        RijndaelAlg.CreateEncryptor(Key, IV),
        CryptoStreamMode.Write);

        // Create a StreamWriter using the CryptoStream.
        StreamWriter sWriter = new StreamWriter(cStream);

        try

```



```

    {
        // Write the data to the stream
        // to encrypt it.
        sWriter.WriteLine(Data);
    }
    catch (Exception e)
    {
        MessageBox.Show( e.Message);
    }
    finally
    {
        // Close the streams and
        // close the file.
        sWriter.Close();
        cStream.Close();
        fStream.Close();
    }
}
catch (CryptographicException e)
{
    MessageBox.Show( e.Message);
}
catch (UnauthorizedAccessException e)
{
    MessageBox.Show( e.Message);
}
}
}

```

و کد زیر مربوط است به دکمه نمایش همراه با تابع آن :

```

private void button2_Click(object sender, EventArgs e)
{
    Rijndael RJ = Rijndael.Create();
    string strRamz = textBox2.Text;
    string FileName = "CText.txt";
    char[] _ch = textBox2.Text.ToCharArray();
    byte[] _by = new byte[32];
    for (int i = 0; i < 32; i++)
    {
        _by[i] = (byte)_ch[i];
    }
    byte[] _byIV = new byte[16];
    for (int i = 0; i < 16; i++)
    {
        _byIV[i] = 0;
    }
    RJ.Key = _by;
    RJ.IV = _byIV;
    richTextBox2.Text = DecryptTextFromFile(FileName, RJ.Key, RJ.IV);
    MessageBox.Show("finish");
}
public string DecryptTextFromFile(String FileName, byte[] Key, byte[]

```

IV)

```

{
    try
    {
        // Create or open the specified file.
        FileStream fStream = File.Open(FileName, FileMode.Open);

        // Create a new Rijndael object.
        Rijndael RijndaelAlg = Rijndael.Create();

        // Create a CryptoStream using the FileStream
        // and the passed key and initialization vector (IV).
        CryptoStream cStream = new CryptoStream(fStream,
            RijndaelAlg.CreateDecryptor(Key, IV),
            CryptoStreamMode.Read);

        // Create a StreamReader using the CryptoStream.
        StreamReader sReader = new StreamReader(cStream);

        string val = null;

        try
        {
            // Read the data from the stream
            // to decrypt it.
            val = sReader.ReadLine();

        }
        catch (Exception e)
        {
            MessageBox.Show(e.Message);
        }
        finally
        {
            // Close the streams and
            // close the file.
            sReader.Close();
            cStream.Close();
            fStream.Close();
        }

        // Return the string.
        return val;
    }
    catch (CryptographicException e)
    {
        MessageBox.Show(e.Message);
        return null;
    }
    catch (UnauthorizedAccessException e)
    {
        MessageBox.Show(e.Message);
        return null;
    }
}
}

```



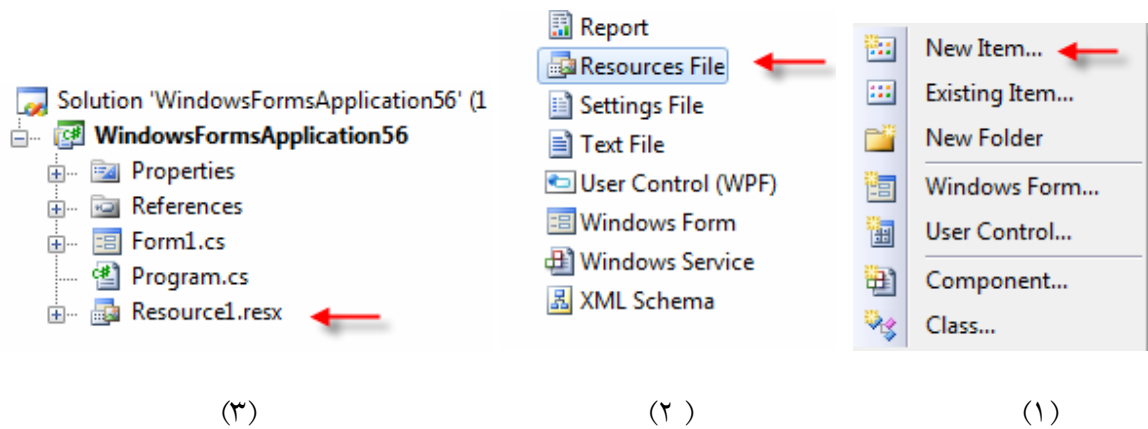
برای مقایسه هم می توانید به شکل کد زیر عمل کنید یعنی رشته کد رمز شده شما با رشته ورودی جدید(که کد می شود) مقایسه می شود و اگر برابر بود مقدار true برگشت داده می شود.

```
bool verifyMd5Hash(string input, string hash)
{
    // Hash the input.
    string hashOfInput = getMd5Hash(input);
    // getMd5Hash تابعی است مانند کد فرم لود بالا
    // Create a StringComparison to compare the
hashes.
    StringComparison comparer =
StringComparison.OrdinalIgnoreCase;

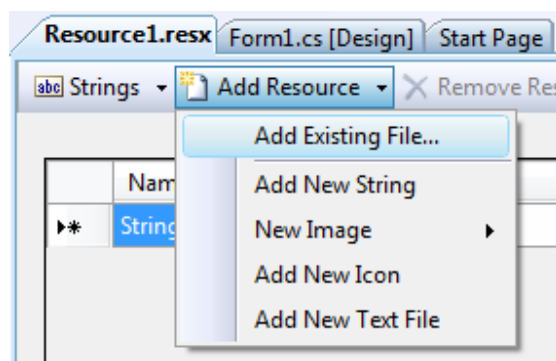
    if (0 == comparer.Compare(hashOfInput, hash))
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

### نحوه اضافه کردن فایلها به برنامه :

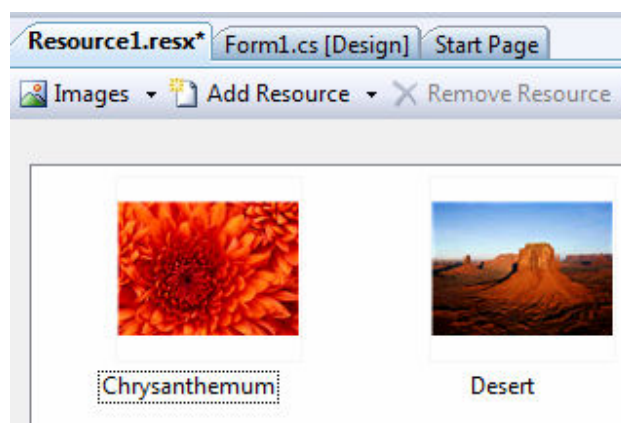
هر وقت خواستید یک فایل به برنامه اضافه کنید تا در موقع لزوم از آن فایل استفاده کنید باید آن را در Resource File ذخیره کنید . وقتی اینکار را انجام بدید فایلهای مورد نظر همراه فایل اجرایی حمل می شود یکی از کاربردهای مهم این کار ساخت فایل های نصب (Setup) است به این صورت که فایلهای اجرایی و دیگر فایلهای همراه برنامه را در یک برنامه دیگر اضافه می کنیم . برای انجام اینکار از Add New Item یک Resource File به برنامه اضافه کنید



بعد از پنجره Resource1.resx فایل های مورد نیاز را از قسمت Add Resource و Add Existing File به برنامه اضافه کنید.



مثلا دو عکس به برنامه اضافه کنید



وقتی اینکار را انجام دادید و برنامه را کامپایل کردید ظرفیت فایل اجرایی برنامتون نصبت به فایلهایی که اضافه کردید افزایش می یابد . حال یک دکمه بر روی فرم قرار دهید (وتکست آن را به نصب تغییر دهید ) تا وقتی بر روی آن کلیک کردید یکی از عکس ها در جایی از حافظه کپی شود :

(نکته : این فایلها به صورت آرایه ای از بایتها در فایل اجرایی ذخیره می شود مگر اینکه برای سی شارپ شناخته شده باشد برای مثال اگر فایل شما عکس بود سی شارپ آن را به صورت عکس تشخیص می دهد در غیر این صورت به صورت آرایه ای از بایتها ذخیره می کند. برای امتحان یک فایل دیگر با فرمت mp3 به برنامه اضافه کنید.)

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        byte[] _MYFile =
WindowsFormsApplication56.Resource1.TRACK_01;
        Bitmap mypic =
WindowsFormsApplication56.Resource1.Chrysanthemum;
        System.IO.FileStream _FS = new
System.IO.FileStream("D:\\T1.mp3", System.IO.FileMode.Create);
        System.IO.BinaryWriter _WR = new System.IO.BinaryWriter(_FS);
        _WR.Write(_MYFile);
        _WR.Close();
        mypic.Save("D:\\mypic.jpg",
System.Drawing.Imaging.ImageFormat.Jpeg);
        MessageBox.Show("finish copy");
    }
    catch { }
}
```

در این قسمت در باره دیگر کلاسهای داخلی System.IO بحث خواهیم کرد .

System.IO.File.Copy : از این متد که در کلاس File قرار دارد برای کپی فایلها از مکانی از حافظه به مکان دیگر استفاده می شود.

System.IO.File.Copy (

▲1 of 2 ▼	void File.Copy (string sourceFileName, string destFileName)
sourceFileName:	The file to copy.

مثال : عکسی را از درایو C به d کپی می کنیم :

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.File.Copy("c:\\pic.jpg", "d:\\pic.jpg");
}
```

System.IO.File.Delete: برای پاک کردن فایل از این متد استفاده می شود. مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.File.Delete("c:\\pic.jpg");
}
```

System.IO.File.Exists: برای فهمیدن اینکه یک فایل وجود دارد یا نه از این متد استفاده می شود اگر فایلی وجود داشت این متد مقدار true بر می گرداند و برعکس.

مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    if (System.IO.File.Exists("c:\\pic.jpg")==true)
    {
        MessageBox.Show("file is exists");
    }
}
```

System.IO.File.Move: برای حرکت دادن یک فایل از جایی به جای دیگر مثل کپی عمل می کند ولی با این تفاوت که فایل اصلی پاک می شود.

نکته: از این متد برای تغییر نام فایلها نیز استفاده می شود!!!! در جلو مثال کلی در این باره حل خواهیم کرد.

System.IO.File.WriteAllText: برای نوشتن در فایلهای متنی استفاده می شود که قبلا مثالی در این باره حل کردیم.

System.IO.Directory.CreateDirectory: برای ایجاد فولدر در یک مسیر از این متد استفاده می شود. مثال:

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.Directory.CreateDirectory("c:\\myfolder\\jk");
}
```

: System.IO.Directory.Delete  
از این متد برای پاک کردن مسیر استفاده می شود.

System.IO.Directory.GetFiles: از این متد برای بدست آوردن فایل‌های درون یک مسیر استفاده می شود مثال:

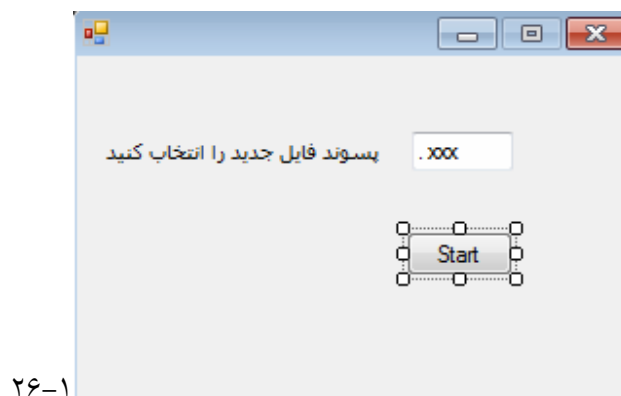
```
private void button1_Click(object sender, EventArgs e)
{
    string[] str= System.IO.Directory.GetFiles("c:\\myfolder\\jk");
    foreach (string i in str)
    {
        listBox1.Items.Add(i);
    }
}
```

همچنین می توانیم برای جستجو نیز از این متد استفاده کنیم مثال: فایل‌هایی .exe در مسیر مشخص شده:

```
private void button1_Click(object sender, EventArgs e)
{
    string[] str=
System.IO.Directory.GetFiles("c:\\Windows\\System32", "*.exe");
    foreach (string i in str)
    {
        listBox1.Items.Add(i);
    }
}
```

مثال:

برنامه ای بنویسید که پسوند تعدادی از فایلها را تغییر دهد در این برنامه هر جا که فایل اجرایی کپی شود تمام فایل‌های کنار این فایل اجرایی به پسوند مشخص شده تغییر داده می شود. برای انجام این برنامه شکل آن را به صورت شکل ۱-۲۶ تغییر دهید:



بعد کد زیر را برای دکمه start بنویسید:

```
{
    string x;
    string io = System.IO.Path.GetFullPath("changeExtention.exe");
```



متد `getfullpath` تمام مسیر را که این فایل اجرایی در آن قرار گرفته همراه با نام آن می دهد مثل: `c:\newfolder\changeExtention.exe` و لی ما می خواهیم فایل های داخلی این مسیر را بدست بیاوریم بنابراین آخرین نام را با استفاده از `replace` حذف می کنیم تا مسیر خالص بدست بیاید مثل `c:\newfolder:`

```
x = io;
io = io.Replace("changeExtention.exe", " ");
```

تمام فایل های (همراه با مسیر کامل) این مسیر بدست می آید و داخل آرایه ذخیره میشود

```
string[] _M = System.IO.Directory.GetFiles(io);
```

تمام فایل ها تغییر نام پیدا می کنند به جز خود فایل اجرایی که داخل این مسیر (پوشه) است.

```
foreach (string str in _M)
{
    if (!str.EndsWith("changeExtention.exe"))
    {
        System.IO.File.Move(str,
System.IO.Path.ChangeExtension(str, textBox1.Text));
    }
}
:System.IO.DriveInfo
```

از این کلاس برای بدست آوردن مشخصات کامل درایوهای کامپیوتر استفاده می شود .

مثال :

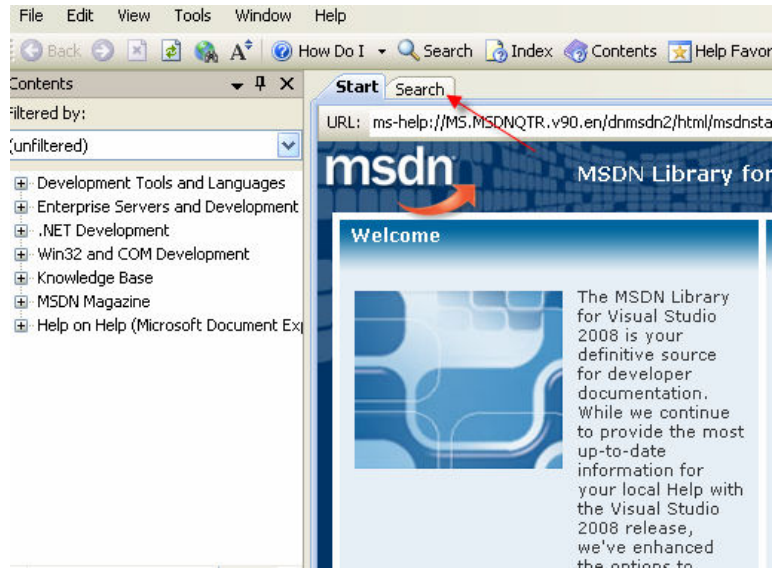
مشخصات درایو (C) کامپیوتر را به دست بیاورید . (نوع فرمت و ظرفیت)

```
System.IO.DriveInfo DF = new System.IO.DriveInfo("C");
listBox1.Items.Add(DF.DriveFormat);
listBox1.Items.Add(DF.TotalSize);
```

اگر خواستید همه مشخصات درایو های کامپیوتر را بدست بیاورید باید به این صور عمل کنید:

```
private void button1_Click(object sender, EventArgs e)
{
    System.IO.DriveInfo[] DR = System.IO.DriveInfo.GetDrives();
listBox1.Items.Add(DR[0].TotalSize.ToString()+DR[0].Name);
listBox1.Items.Add(DR[3].TotalSize.ToString() + DR[3].Name);
}
```

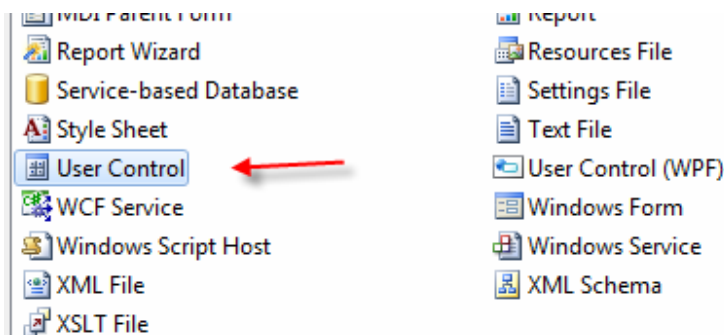
کتابخانه های سی شارپ بسیار وسیع هستند و برای فهمیدن همه آنها باید مراجعه کنید به کتابخانه ام اس دی ان سی شارپ که موقع نصب ویژوال استدیو باید آن را نیز نصب کنید در این کتابخانه با جستجو درباره موضوع مورد نظر می توانید به نحوه استفاده از آنها پی ببرید.



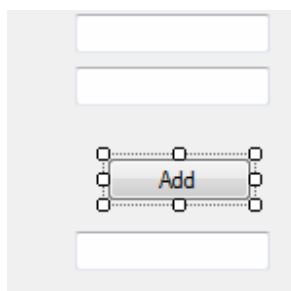
:UserController

فرض کنید چندین فرم در یک برنامه استفاده کردید و در هر فرم از چند کنترل مشترک استفاده شده است مثلاً هر فرم نیاز دارد که چند عدد را با هم جمع کند و این در همه فرمها مشترک است ولی باید برای همه فرمها کدهای جمع دو عدد و کنترل های مورد نیاز قرار داده شود که اگر هزار فرم داشته باشید باید هزار تا از این کنترل ها درست کنید که این کار اصلاً با صرفه نیست بنابراین باید راه حل دیگری برای اینکار استفاده شود که آن روش استفاده از UserControl است که یکی از آن درست می کنیم و در همه صفحات از آن استفاده می کنیم .

برای کار با UserControl کافیه از قسمت New Items آن را به برنامه add کنید .



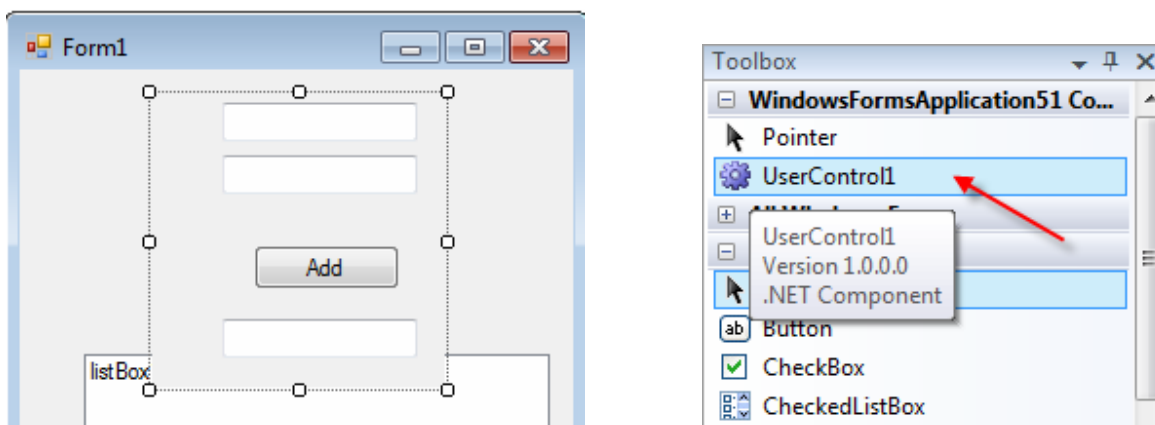
بعد کنترل های مورد نیاز را روی آن قرار دهید مطابق شکل زیر:



برای مثال کد جمع دو عدد را برای دکمه بنویسید:

```
private void Add_Click(object sender, EventArgs e)
{
    int i = int.Parse(textBox1.Text);
    int j = int.Parse(textBox2.Text);
    textBox3.Text = (i + j).ToString();
}
```

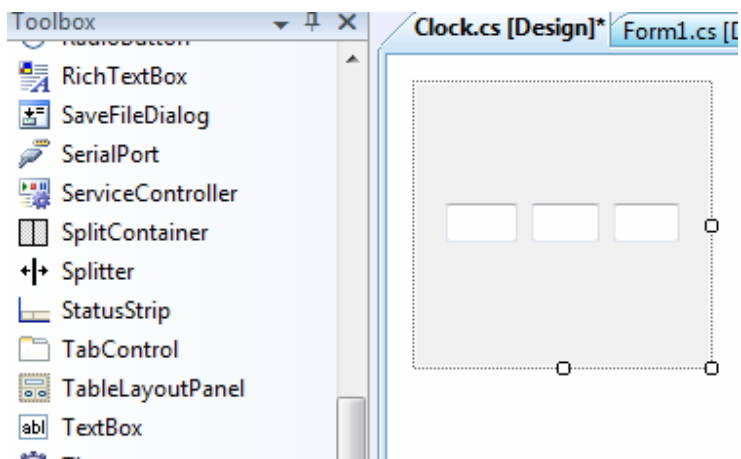
بعد برنامه را Build کنید تا user control ساخته شود اگر بدون خطا کامپایل شود یک شکل با نام user control در نوار ابزار ظاهر می شود. بعد آن را بکشید و بر روی فرمهایتان قرار دهید و بعد برنامه را اجرا کنید و نتیجه را ببینید.



مثال : برنامه ساعت را با user control بنویسید و در صفحات قرار دهید:

مثل تمرین قبلی یک user control با نام clock به برنامه اضافه کنید و سه تکست باکس و یک

تایمر بر روی فرم قرار دهید

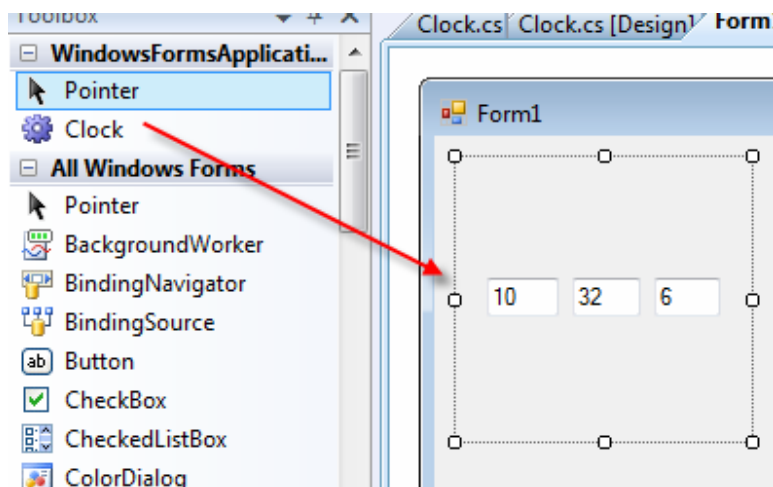


و کد زیر را برای load و تایمر بنویسید: (در load تایمر فعال می شود)

```
private void timer1_Tick(object sender, EventArgs e)
{
    textBox1.Text = DateTime.Now.Hour.ToString();
    textBox2.Text = DateTime.Now.Minute.ToString();
    textBox3.Text = DateTime.Now.Second.ToString();
}

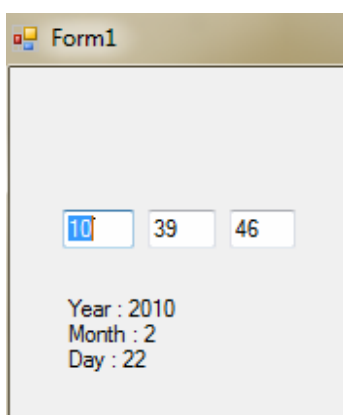
private void Clock_Load(object sender, EventArgs e)
{
    timer1.Start();
}
```

حال برنامه را build کنید و یک ساعت بر روی فرمهایتان قرار دهید:



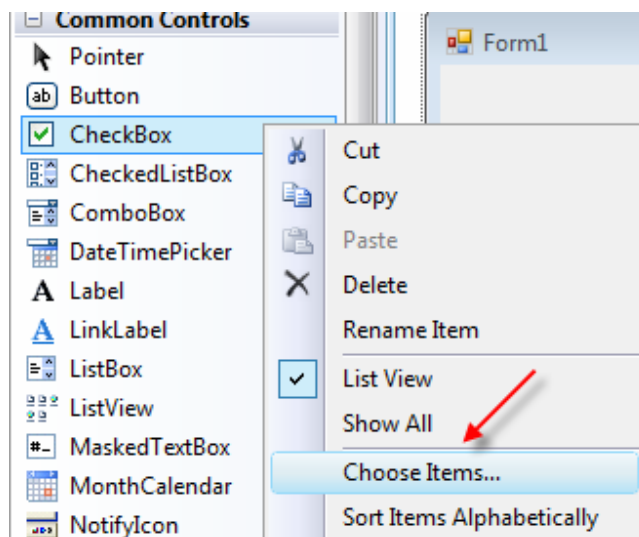
روش کار به این صورت است که کلاس `DateTime` مشخصات کامل زمان سیستم را بر می گرداند ولی برای اینکه در هر لحظه زمان را بدست بیاوریم یک تایمر قرار می دهیم که کار تایمر اینست که در هر ۱۰۰ میلی ثانیه اجرا می شود (می توانید زمان اجرا را تغییر دهید) تا اگر زمان و یا تاریخ تغییر کرد آن را نمایش دهد. حتی می توانید تاریخ را نیز به این برنامه اضافه کنید مثل کد زیر: (سه label بر روی فرم قرار دهید)

```
private void timer1_Tick(object sender, EventArgs e)
{
    textBox1.Text = DateTime.Now.Hour.ToString();
    textBox2.Text = DateTime.Now.Minute.ToString();
    textBox3.Text = DateTime.Now.Second.ToString();
    label1.Text = "Year : " + DateTime.Now.Year.ToString();
    label2.Text = "Month : " + DateTime.Now.Month.ToString();
    label3.Text = "Day : " + DateTime.Now.Day.ToString();
}
```

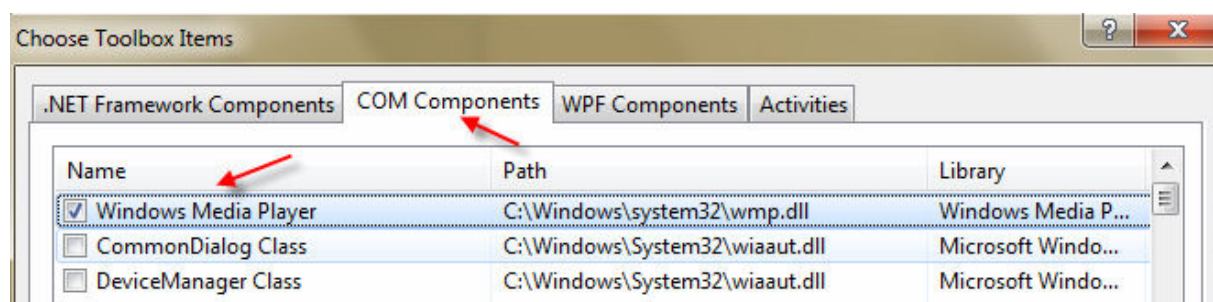




Windows Media Player in Csharp

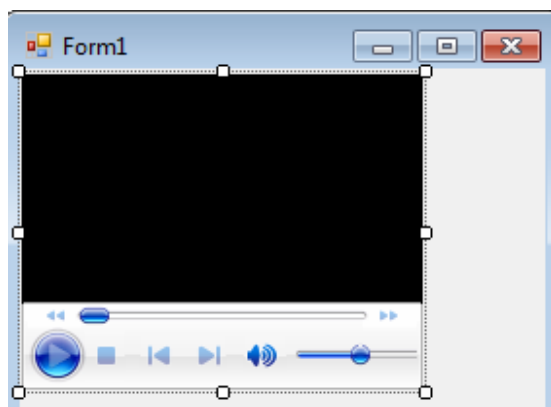
گاهی اوقات نیاز دارید که در برنامه آهنگی برای کاربر پخش کنید برای اینکار می توانید از مدیا پلیر ویندوز استفاده کنید. برای انجام اینکار در نوار ابزار کلیک راست کرده و `choose items` را انتخاب کنید



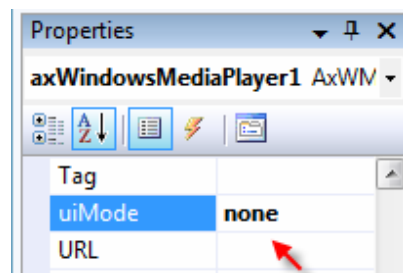
بعد مدیا پلیر را انتخاب کنید و دکمه `ok` را فشار دهید



بعد یک آیکن در نوار ابزار   ظاهر می شود آن را بکشید و بر روی فرم قرار دهید

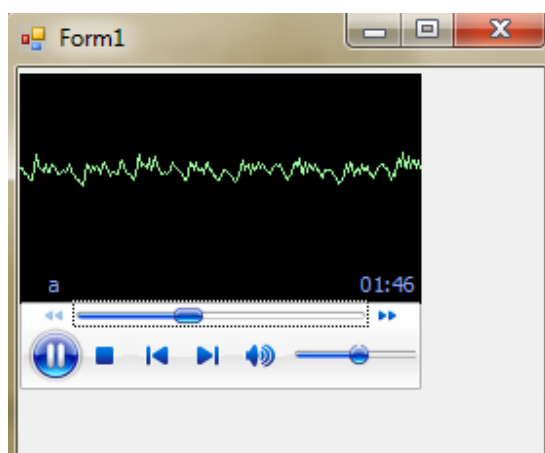


حال اگر خواستید نوار ابزار مدیا پلیر نمایش داده نشود می توانید `uiMode` آن را به `none` تغییر دهید و اگر خواستید هیچ چیز نمایش داده نشود `Visible` آن را `false` کنید.



حال کد زیر را در `load` فرم بنویسید تا آهنگهای دلخواه شما را موقع بالا آمدن فرم به ترتیب پخش کند برای اینکار آرایه ای از `string` ایجاد کنید و نام آهنگ های خود را در آرایه قرار دهید و خود آهنگ را کنار برنامه قرار دهید و به لیست پخش مدیا پلیر اضافه کنید و پخش کنید مانند کد زیر:

```
private void Form1_Load(object sender, EventArgs e)
{
    string[] strNameof = new string[2];
    strNameof[0] = "a.mp3";
    strNameof[1] = "b.mp3";
    for (int i = 0; i < strNameof.Length; i++)
    {
        axWindowsMediaPlayer1.currentPlaylist.appendItem(axWindowsMediaPlayer1.newMedia(strNameof[0]));
    }
    axWindowsMediaPlayer1.Ctlcontrols.play();
}
```



حتی می توانیم فایل های ویدیوی را نیز پخش کنید .

نکته : مزایای مدیا پلیر اینست که تعداد فایل های بیشتری را می تواند پخش کند ولی دارای مشکلاتی است از جمله اینکه باید در سیستم عامل برنامه windows media player نصب باشد وگرنه برنامه ای که نوشتید کار نمی کند.

اگر خواستید از مدیا پلیر استفاده نکنید و فقط فایل های صوتی با فرمت (wav) را پخش کنید می توانید از کلاس `System.Media.SoundPlayer` استفاده کنید که جز کلاس های خود سی شارپ است و نیازی به نصب مدیا پلیر نیست روش کار این کلاس به صورت کد زیر است :

```
private void Form1_Load(object sender, EventArgs e)
{
    System.Media.SoundPlayer my = new
System.Media.SoundPlayer("a.wav");
    my.Play();
}
```

اگر خواستید با دکمه ای می توانید پخش آهنگ را متوقف کنید به شرطی که کلاس را عمومی تعریف کنید.

```
System.Media.SoundPlayer my = new System.Media.SoundPlayer("a.wav");

private void Form1_Load(object sender, EventArgs e)
{
    my.Play();
}

private void button1_Click(object sender, EventArgs e)
{
    my.Stop();
}
```





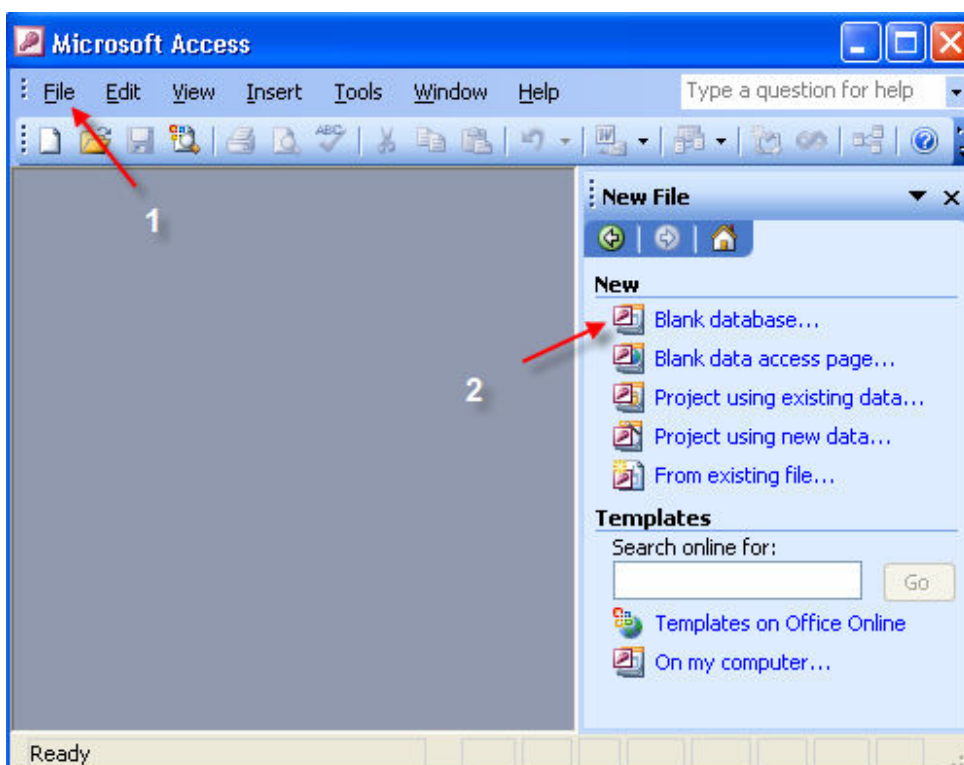
**فصل دوم :**  
**پایگاه داده ها**

امروزه نیاز به ذخیره سازی اطلاعات و عملیات روی آنها از اهمیت بالایی برخوردار است به همین منظور زبان های برنامه نویسی از جمله سی شارپ برای راحتی کار عملیات لازم را برای ذخیره سازی اطلاعات روی بانک اطلاعاتی رافراهم آورده اند.

از جمله بانکهای اطلاعاتی که امروزه مورد استفاده قرار میگیردمی توان به اکسس و اوراکل اشاره کرد که دارای سازمان دهی مناسب و امنیت بالا برای ذخیره سازی اطلاعات هستند در این بخش ما خواستیم تا با استفاده از سی شارپ این عملیات را بر روی پایگاه داده اکسس انجام دهیم اما قبل اینکار کمی با پایگاه داده اکسس آشنا می شویم.

## ۱-۲ محیط اکسس

برای ایجاد یک بانک اطلاعاتی در اکسس ۲۰۰۳ کفایت مطابق شکل ۱-۲ عمل کنید.

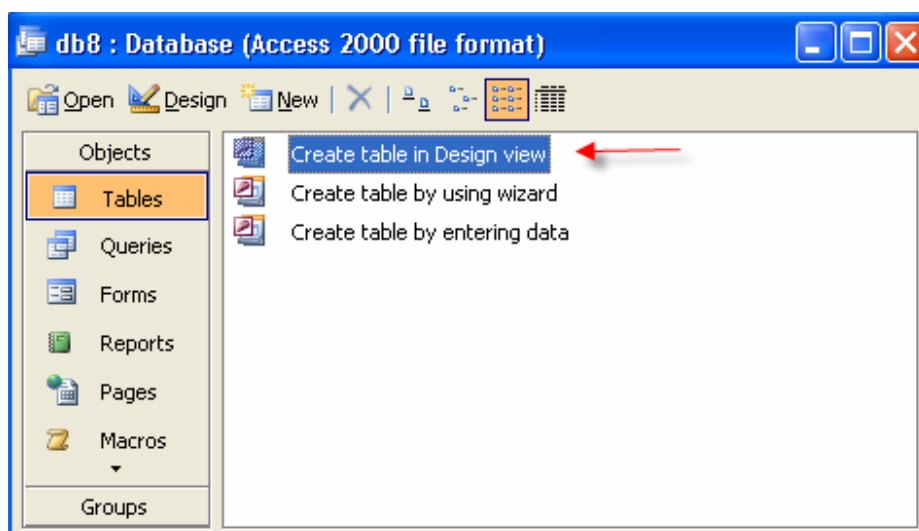


شکل ۱-۲

بعد از این کار از شما پرسیده می شود که نام بانک اطلاعاتی و محل ذخیره سازی آن را مشخص کنید یک نام برای آن انتخاب کنید و مسیر ذخیره سازی آن را مشخص کنید. نوبت آن رسیده که جداول را ایجاد کنیم برای این کار کفایت در پنجره باز شده بر روی گزینه

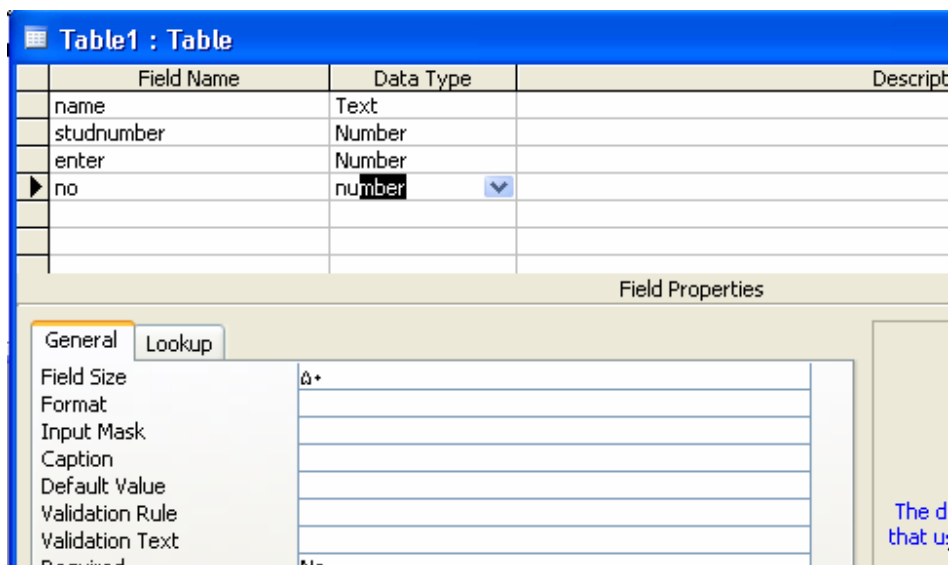
### شکل ۲-۲ Create table in design view

کلیک کنید. با کلیک کردن بر روی این گزینه پنجره ای باز میشود که در آن به طراحی جداول بانک اطلاعاتی می پردازیم برای این کار کفایت فیلد های جداول بانک اطلاعاتی و نوع آن را مشخص کنیم مثلا در سیستم بانک اطلاعاتی دانشگاه فیلد های مانند نام دانشجو شماره دانشجو نمره و سال ورودی دانشجو مطرح می شود در این فیلد های شماره نمره و سال ورودی دانشجو بهتر است از نوع صحیح **int** انتخاب شود و برای نام بهتر است از نوع رشته انتخاب شود.



شکل ۲-۲

همانطور که در شکل ۲-۳ مشاهده می کنید ما فیلدهای جدول را تنظیم کردیم بعد از تنظیم فیلدها کفایت پنجره را ببندید. موقع بستن پنجره از شما پرسیده می شود که یک نام برای جدول خود انتخاب کنید این کار را انجام بدید و خارج شوید .



شکل ۲-۳

اگر برنامه‌تون به جداول دیگری نیاز داشت می‌توانید به همین طریق عمل کنید و جدول دیگری بسازید حالا نام جدول شما به پنجره اضافه شد برای مشاهده جدول کفایت بر روی آن کلیک کنید. در اول کار جدول شما خالی است چون هنوز اطلاعاتی در آن وارد نشده است از همینجا می‌توانید اطلاعاتی برای جدول خود وارد کنید و تغییرات را مشاهده کنید اما هدف ما این نیست که از همینجا اطلاعات را وارد کنیم بلکه ما می‌خواهیم از طریق سی شارپ و محیط ویژوال عملیات لازم را انجام دهیم که این کار نیاز به دانستن زبان اکسس دارد بنابراین قبل از وارد شدن به این بحث که در فصل بعدی توضیح خواهیم داد ابتدا با اکسس آشنا می‌شویم چون در محیط ویژوال نیز بیشتر با زبان اکسس کار خواهیم کرد.

## ۲-۲ دستورات اکسس

حالا توضیحاتی درباره اکسس ارایه می‌دیم در پنجره اکسس محیطی وجود دارد که در آن می‌توانیم زبان اکسس را پیاده‌سازی کنیم برای مشاهده این محیط کفایت گزینه

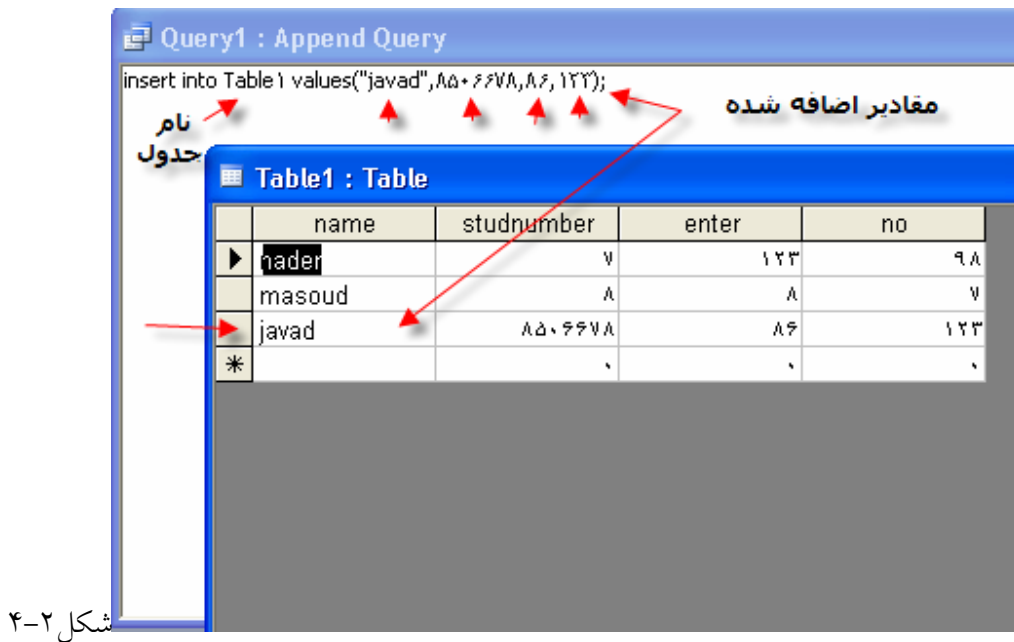
Create query by using wizard

را انتخاب کنید و زبان اکسس را در آن پیاده‌سازی کنید. حالا با زبان اکسس (دستورات اکسس آشنا) می‌شویم.

INSERT

از این دستور برای اضافه کردن دیتا به جدول دیتابیس استفاده می شود. مثلا با نوشتن دستور زیر مقادیر ذکر شده در دستور به جدول اضافه می شود [1] مطابق شکل ۲-۴.

```
insert into Table1 values"javad",8506678,86,123;
```



شکل ۲-۴

## DELETE

از این دستور برای پاک کردن اطلاعات از جدول استفاده می شود [1]. مثلا برای پاک کردن سطری از جدول که نام آن جواد است استفاده می شود.

```
delete from Table1 where name="javad";
```

در ضمن همانطور که در دستور بالا مشاهده فرمودید می توانید شرط نیز به دستوراتمان اضافه کنیم مثلا بگوییم سطرهایی از جداول را حذف کن که نام آن جواد است. به این نکته نیز توجه کنید که اگر در بعضی

از دستورات شرطی وارد نکنیم کل سطرها را مورد بررسی قرار میدهند مثلا اگر در دستور بالا شرط نبود کل سطرها را پاک می کرد.

#### UPDATE

از این دستور برای تغییر در جداول استفاده می شود [1] مثلا اگر خواستیم شماره دانشجویی شخصی را عوض کنیم از این دستور استفاده می کنیم .

```
update Table1 set studnumber=123321 where name="masoud";
```

در این جا هم به این نکته باید توجه داشته باشیم که اگر عنصری را که می خواهیم مقدار آن را تغییر دهیم به عنوان کلید انتخاب شده باشد نمی توانیم مقدار آن را تغییر دهیم.

#### SELECT

از این دستور برای انتخاب ستون هایی از جدولمان استفاده می شود مثلا با نوشتن دستور زیر ستون های نام دانشجو انتخاب می شود

```
SELECT name
```

```
FROM Table1;
```

این ها دستوراتی بودند که در بیشتر برنامه ها از آن ها به عنوان دستورات اصلی استفاده می شود البته دستورات جالب دیگری نیز وجود دارد که می توانید از آنها استفاده کنید . برای فهمیدن آنها می توانید به **help** اکسس مراجعه کنید .

اینها توضیحات مختصری بود که درباره پایگاه داده اکسس ارایه کردیم البته موقعی که از این دستورات در برنامه استفاده کردیم بیشتر با آنها آشنا خواهید شد .

## فصل سوم

### آشنایی با کلاس های دیتابیس در سی شارپ

#### مقدمه

در این فصل می خواهیم در مورد کلاس های دیتابیس در سی شارپ بحث کنیم. در سی شارپ کلاسهای وجود دارد که با استفاده از آنها می توانیم به پایگاه داده اکسس یا پایگاه داده های دیگر از جمله SQL وصل شویم. وظیفه این کلاسها این است که دستورات ما را به پایگاه داده مورد نظر انتقال داده و عملیات روی آنها را پیاده سازی نماید .

مثلا اگر خواستیم دستور **insert** را که در فصل قبل آن را توضیح دادیم بر روی پایگاه داده مورد نظر پیاده سازی کنیم از این کلاس ها استفاده می کنیم .

### ۱-۳ هدر System.Data.OleDb

همانطور که گفتیم کلاس هایی وجود دارد که عملیات وصل شدن به پایگاه داده مورد نظر را انجام می دهد. برای وصل شدن به پایگاه داده اکسس کلاس هایی وجود دارد که در این هدر **System.Data.OleDb** جا گرفته اند.

برای وصل شدن به پایگاه داده **sql** نیز می توانید از هدر **System.Data.SqlClient** استفاده کنید. برای استفاده از این کلاس ها کفایت آنها را به هدرها اضافه کنید [3] مطابق شکل ۱-۳.

```
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb;
using System.Data.SqlClient;
namespace WindowsFormsApplication75
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

شکل ۱-۳ هدرهای لازم که به برنامه اضافه شده است

حال به بررسی کلاس های داخلی این هدرها می پردازیم.

### ۲-۳ OleDbConnection

از این کلاس برای وصل شدن به پایگاه داده استفاده می شود یعنی قبل از این که دستورات را بنویسیم باید به پایگاه داده وصل شویم وقتی این عمل اتفاق افتاد زمان فرستادن دستورات می رسد که برای این کار نیز کلاس

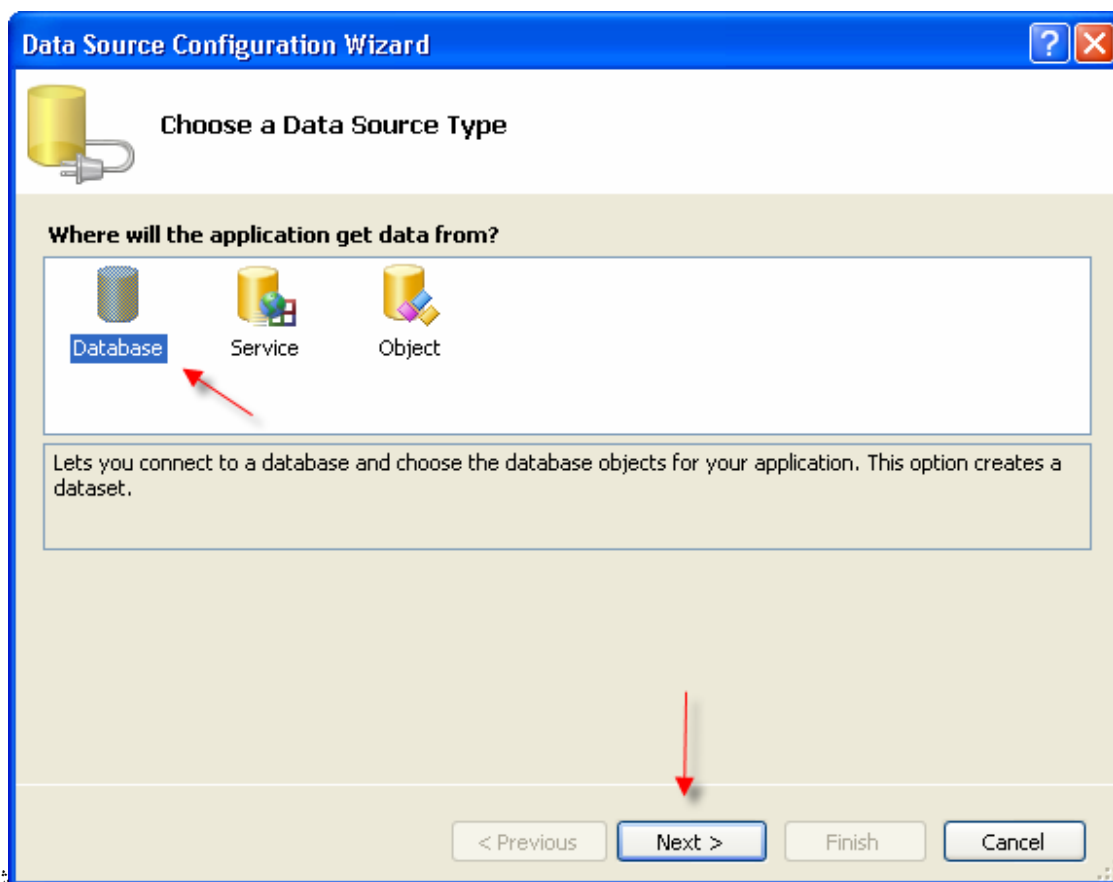


دیگری وجود دارد **OleDbCommand** که این کار را انجام می دهد که در قسمت بعدی درباره آن بحث خواهیم کرد. حال برگردیم به کلاس **oledbconnection**. این کلاس برای وصل شدن به پایگاه داده نیاز به یک **ConnectionString** است این رشته شامل نام دیتابیس ، مسیر و رمز آن است در زیر یک نمونه از آن آورده شده است که مربوط می شود به پایگاه داده اکسس:

```
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source="C:\db8.mdb";password="456"
```

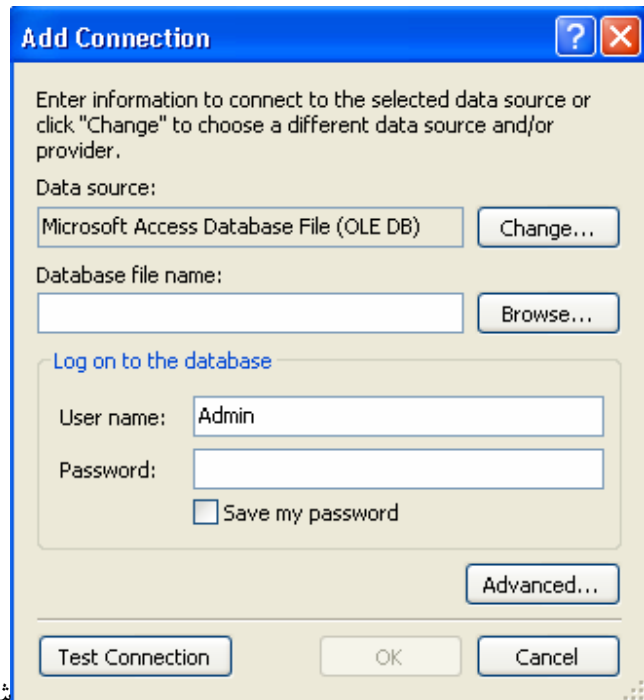
در قسمت **Microsoft.Jet.OLEDB.4.0** مشخص می شود که نوع اکسس من از چه نوعی است مثلا ۲۰۰۳ است یا ۲۰۰۷ که با توجه به نوع آن متفاوت است در قسمت **source** به مسیر دیتابیس ایجاد شده اشاره می کند و قسمت آخر نیز مربوط می شود به امنیت پایگاه داده .

اما شاید این سوال پیش آید که ما از کجا بفهمیم کدامین رشته مال کدامین دیتابیس است در جواب باید بگویم که روشهای زیادی برای بدست آوردن این رشته وجود دارد در زیر به یکی از این روش ها اشاره می کنیم . فرض کنید یک دیتابیس اکسس ایجاد کردید و درجایی از حافظه ذخیره کردید حال یک پروژه سی شارپ ایجاد کنید و از منو **data** بر روی گزینه **add new data source....** کلیک کنید حال پنجره ای باز می شود در این پنجره مطابق شکل ۳-۲ بر روی گزینه **database** را کلیک کنید بعد **next** را بزنید .

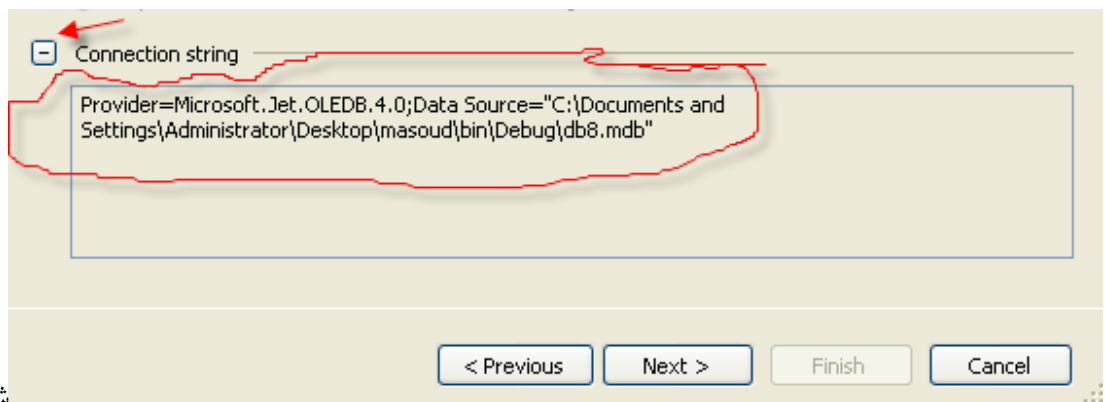


شکل ۳-۲

بعد در پنجره جدید بر روی گزینه **new connection** کلیک کنید تا پنجره ای مطابق شکل ۳-۳ باز شود . در این پنجره می توانید پایگاه داده مورد نظر خود را از قسمت **change** انتخاب کنید وقتی بر روی این گزینه کلیک کردید پنجره دیگری باز می شود که در این پنجره اگر خواستید به پایگاه داده اکسس وصل شود باید بر روی گزینه **Microsoft Access Database File** کلیک کنید و اگر خواستید به **sql** وصل شوید باید بر روی گزینه **Microsoft SQL Server** کلیک کنید وقتی اکسس را انتخاب کردید در پنجره **Add connection** بر روی گزینه **browse** کلیک کنید و از محلی که فایل دیتابیس را در آن ذخیره کردید فایل دیتابیس را انتخاب کنید اگر رمزی به فایل دیتابیس دادید در قسمت **password** رمز آن را وارد کنید بعد بر روی دکمه **ok** کلیک کنید بعد در صفحه ظاهر شده می توانید **ConnectionString** را مشاهده کنید مطابق شکل ۳-۴ . آن را کپی کنید و در سی شارپ استفاده کنید . برای استفاده از آن بهتر است یک متغیر از نوع **String** تعریف کنید و این رشته را در آن ذخیره کنید [4] تا در برنامه از آن استفاده کنید.



شکل ۳-۳ صفحه انتخاب نوع پایگاه داده



شکل ۴-۳

بهرتر است این کار را بر روی پایگاه داده های دیگر نیز انجام دهید و تفاوت را مشاهده فرمایید.

حال ببینیم از این کلاس `oledbconnection` چگونه استفاده می شود. برای استفاده از این کلاس کفایت یک شی از آن مطابق زیر ایجاد شود و `connectionString` آن را برابر با رشته ای قرار بدید که روش بدست آوردن آن را در قسمت قبل توضیح دادم .

```

string str = @"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\db8.mdb";
OleDbConnection co = new OleDbConnection();
co.ConnectionString = str;

```

در ضمن این کلاس دارای دو متد دیگری نیز برای باز کردن دیتابیس و بستن آن دارد که لازم است هر وقت خواستیم دستوری به دیتابیس بفرستیم ابتدا آن را باز کنیم و بعد از انجام کار آن را ببندیم تا برای استفاده های دیگر آن را آزاد کنیم. این دو دستور به شرح زیر هستند.

```

co.Open;
//insert code
co.Close;

```

### OleDbCommand 3-3

همانطور که در قسمت قبلی گفتیم برای فرستادن دستورات به پایگاه داده کلاسی وجود دارد که این کار را انجام می دهد نام این کلاس **OleDbCommand** است نحوه استفاده از این کلاس به این صورت است که یک شی از آن ایجاد می کنیم و این شی را به کلاس **oledbconnection** منتصب می کنیم بعد دستوراتمان را می فرستیم مطابق کد زیر که در آن نحوه نوشتن دستور **insert** را توسط این کلاس نشان می دهد.

```

string str = @"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\db8.mdb";
OleDbConnection co = new OleDbConnection();
co.ConnectionString = str;
string str2="insert into Table1 values('nader',42,87,12300)";
OleDbCommand cmd = new OleDbCommand(str2, co);
co.Open();
cmd.ExecuteNonQuery();
co.Close();

```

نکته ای که باید به آن اشاره کنیم این است که باید حتما از متد **ExecuteNonQuery** استفاده کنیم تا تغییرات ما ذخیره شود.

### OleDbDataAdapter 3-4

از این کلاس بیشتر برای پر کردن جدول `DataTable` استفاده می شود. برای استفاده از این کلاس نیز کافیت یک شی از آن ایجاد کنید و متد `Fill` آن را فراخوانی کنیم [3] مانند کد زیر:

```
OleDbDataAdapter odba = new OleDbDataAdapter();
odba.Fill(dt);
```

خطایی که در کد بالا رخ می دهد اینست که `OleDbDataAdapter` نمی فهمد که `datatable` را با چه چیزی پر کند برای اینکار باید آن را به کلاس `oledbcommand` منتصب کنیم تا بفهمد با چه دستوری جدولمان را پر کند مثلاً در دستور زیر جدول دیتابیس را با سطرهایی پرمی کند که شماره دانشجویی آنها بزرگتر از ۸۵ باشد

```
string str = @"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\db8.mdb";
OleDbConnection co = new OleDbConnection();
co.ConnectionString = str;
string str2 = "select * Table1 where (studnumber>85)";
OleDbCommand cmd = new OleDbCommand(str2, co);
co.Open();
cmd.ExecuteNonQuery();
co.Close();
OleDbDataAdapter odba = new OleDbDataAdapter(cmd);
odba.Fill(dt);
```

### DataTable 3-5

برای اینکه ما بتوانیم یک کپی از جدول دیتابیس در سی شارپ داشته باشیم از این کلاس استفاده می کنیم این کلاس به صورت آرایه دوبعدی عمل می کند و ما می توانیم به راحتی و با سرعت زیاد به عناصر آن دستیابی داشته باشیم. از کاربردهای دیگر این کلاس این است که ما میتوانیم جدول `dataGridView` را نیز با آن پر کنیم برای این کار کافیت یک `dataGridView` را در صفحه نمایش قرار دهیم شکل ۳-۵ و کد زیر را بنویسیم

```
//Filldt
dataGridView1.DataSource = dt;
```

این ها کلاس هایی بودند که عملیات لازم برای انجام یک ارتباط ساده با پایگاه داده را انجام می دهند البته چند تا کلاس ساده دیگر نیز وجود دارند که در فصل بعد در موقع استفاده در برنامه آنها را توضیح خواهیم داد.

	name	studnumber	enter	no
▶	محمد	123321	8	76
*				

شکل ۳-۵ پرکردن دیتاگرید

نکته : اگر خواستید با پایگاه داده **sql** کار کنید فقط کافایت نام کلاس ها را به شرح زیر تغییر دهید:

```
// SqlConnection
// SqlCommand
// SqlDataAdapter
```

ومثل مثال های بالا عمل کنید

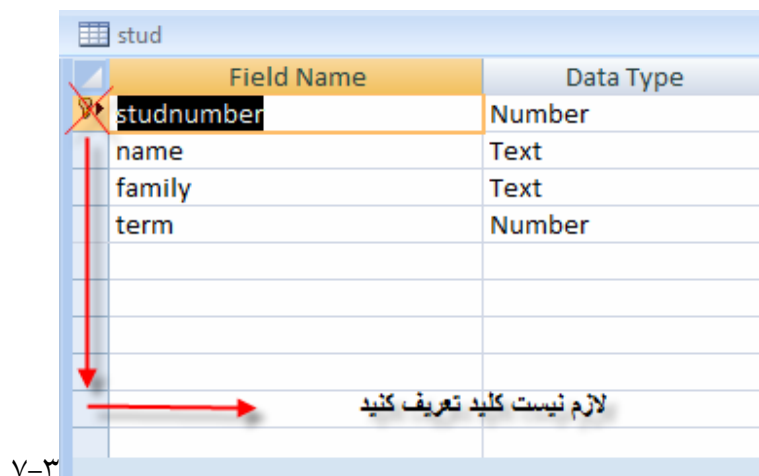
مثال : برنامه ای بنویسید که مشخصات دانشجویان را در یک جدول دیتابیس ذخیره کند :

برای انجام اینکار شکل ظاهری برنامه را مطابق شکل ۳-۶ زیر تنظیم کنید . نحوه کارکرد این برنامه به این صورت است که وقتی دانشجوی جدیدی وارد دانشگاه می شود کاربر با زدن دکمه **new** اجازه می دهد تا مشخصات دانشجوی جدید که در تکست باکسها وارد شده به جدول دیتابیسمان اضافه شود همچنین می توانید مشخصات دانشجویان را تصحیح و یا پاک کنید . همچنین لیست افراد در یک جدول دیتاگرید در زیر فرم آورده شده است .



۶-۳

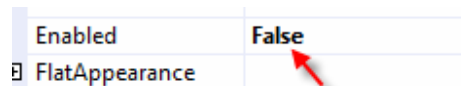
در مرحله کد نویسی باید جداولمان را درست کنیم برای اینکار یک دیتابیس با نام **project.mdb** ایجاد کنید و یک جدول درون این دیتابیس با فیلدهای شماره دانشجویی نام و نام خانوادگی و ترم ورودی دانشجو مطابق شکل ۳-۷ درست کنید



۷-۳

بعد جدول را بانام stud ذخیره کنید تا در محیط سی شارپ با این نام جدول ما فراخوانی شود اگر خواستید می توانید نامهای ذکر شده در بالا را به دلخواه تغییر دهید. بعد از ساخت دیتابیس آن را به کنار فایل اجرایی کپی کنید:

در ابتدا دکمه با تکست Insert را غیر فعال کنید تا موقع فشار دکمه new فعال شود



حال کدهای مربوط به دکمه new را می نویسیم که کار خاصی انجام نمی دهد فقط رشته های داخل تکست باکس را در صورت وجود پاک می کند وبعد کلاس های دیتابیس را تعریف می کنیم

```
private OleDbCommand cmd;
private OleDbConnection con;
private OleDbDataAdapter da;
private DataTable dt;
private string strcon = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=|DataDirectory|\" + @"\project.mdb";
private void button4_Click(object sender, EventArgs e)
{
    textBox1.Clear();
    textBox2.Clear();
    numericUpDown1.Value = 85;
    textBox4.Clear();
    button1.Enabled = true;
}
```

```
private void button1_Click(object sender, EventArgs e)
```



```
{
```

```
con = new OleDbConnection();  
con.ConnectionString = strcon;  
cmd = new OleDbCommand();  
cmd.Connection = con;
```

روش دیگر برای اضافه کردن و (یا انجام دیگر عملیات) که در اینجا مستقیم اضافه نمی کنیم بلکه با استفاده از پارامترها این کار را انجام می دهیم از جمله مزایای این روش اینست که می توانیم با این روش حتی یک عکس نیز به پایگاه داده اضافه کنیم !!!

```
cmd.CommandText = "INSERT INTO stud  
(studnumber,name,family,term) VALUES (@studnumber,@name,@family,@term)";  
  
cmd.Parameters.Add(new OleDbParameter("@studnumber",  
OleDbType.Numeric)).Value = Convert.ToInt32(textBox4.Text);  
cmd.Parameters.Add(new OleDbParameter("@name",  
OleDbType.Char, 50)).Value = textBox1.Text;  
cmd.Parameters.Add(new OleDbParameter("@family",  
OleDbType.Char, 50)).Value = textBox2.Text;  
cmd.Parameters.Add(new OleDbParameter("@term",  
OleDbType.Numeric)).Value = numericUpDown1.Value;  
con.Open();  
cmd.ExecuteNonQuery();  
con.Close();  
}
```

قبلا از اینکه به سراغ کدهای بعدی بریم این کد را بهتر می کنیم یعنی چند شرط خوب به این برنامه اضافه می کنیم تا اولاً موقع اضافه کردن اطلاعات تکست باکس هامون خالی نباشد که با کد زیر حل می شود:

```
if (textBox4.Text != "" && textBox2.Text != "" && textBox1.Text != "")  
{  
    ////////////////همه کدهای بالا که ننوشتیم  
}
```

و نیز نباید کاربر شماره دانشجویی تکراری وارد کند چون شماره دانشجویی منحصر به فرد است که برای انجام اینکار یک تابع با نام Myselect می نویسیم که در آن شماره دانشجویی جدید بررسی می شود تا اگر تکراری نبود برگشت بدهد

```
if (MySelect("select " + " " + "studnumber" + ",name" + ",family" + ",term" +  
" " + "from stud" + " " + "where studnumber=" + textBox4.Text) == false)  
{
```

روش کار تابع به این صورت است که یک رشته از نوع دستورات اکسس می گیرد و عملیات روی آن را انجام می هد مثلا در اینجا برای فهمیدن تکراری نبودن به این شکل عمل کردیم که اگر شماره دانشجویی که کاربر وارد می کند در جدول بود آن را انتخاب می کند و در یک `DataTable` قرار می دهد بنابراین اگر `DataTable` ما پر شد پس حتما این شماره دانشجویی قبلا وجود داشته و مقدار `true` برگشت داده می شود در غیر این صورت این شماره دانشجویی وجود نداشته و مقدار `false` برگشت داده می شود. کد تابع به این صورت است:

```
public bool MySelect(string sql)
{
    con = new OleDbConnection();
    con.ConnectionString = strcon; ;
    cmd = new OleDbCommand();
    cmd.Connection = con;
    da = new OleDbDataAdapter(cmd);
    dt = new DataTable();

    con.Open();
    cmd.CommandText = sql;
    try
    {
        da.Fill(dt);
    }
    catch { MessageBox.Show("Error"); }
    if (dt.Rows.Count > 0)
    {
        return true;
    }
    else
        return false;
}
}
```

کد های مربوط به دکمه پاک کردن مشخصات دانشجو (Delete). روش کار به این صورت سطرهایی از جدولمان را پاک می کند که شماره دانشجویی آن برابر با شماره دانشجویی باشد که کاربر مشخص کرده یعنی عمل پاک کردن به وسیله شماره دانشجویی مشخص می شود ولی قبل از این کار به وسیله یک پیغام از کاربر پرسیده می شود که آیا می خواهد پاک کند و یا نه!!

```
private void button2_Click(object sender, EventArgs e)
{
```

```

try
{
    if (textBox4.Text != "")
    {
        if (MessageBox.Show("Are you want to delete", "?",
MessageBoxButtons.YesNo, MessageBoxIcon.Question,
MessageBoxDefaultButton.Button2) == DialogResult.Yes)
        {
            con = new OleDbConnection();
            con.ConnectionString = strcon;
            cmd = new OleDbCommand();
            cmd.Connection = con;
            con.Open();
            cmd.CommandText = "delete from stud where
studnumber=" + textBox4.Text;
            cmd.ExecuteNonQuery();
            con.Close();
            textBox1.Clear();
            textBox2.Clear();
            textBox4.Clear();
            studAtr_Load(sender, e);
            MessageBox.Show("delete");
        }
    }
    else
    {
        MessageBox.Show("پاک کردن بر اساس شماره دانشجو");
    }
}
catch
{
    MessageBox.Show("Error");
}
}

```

کدهای مربوط به عمل به روز رسانی (دکمه Update) که عمل اصلاح (به روز رسانی) با شماره دانشجویی امکان پذیر است .

```

private void button3_Click(object sender, EventArgs e)
{
    try
    {
        if (textBox4.Text != "")
        {
            con = new OleDbConnection();
            con.ConnectionString = strcon;
            cmd = new OleDbCommand();
            cmd.Connection = con;
            con.Open();
            cmd.CommandText = "update stud set name='" +
textBox1.Text + "',family='" + textBox2.Text

```

```

+""+",term="+numericUpDown1.Value.ToString()+ " where studnumber=" +
textBox4.Text;

        cmd.ExecuteNonQuery();
        con.Close();
        studAtr_Load(sender, e);
        MessageBox.Show("set update..");
    }
    else
    {
        MessageBox.Show(" دانشجو شماره اساس بر درآوردن روز به ");
    }
}
catch
{
    MessageBox.Show("Error : please set your data");
}
}
}

```

کد دکمه خروج:

```

private void button6_Click(object sender, EventArgs e)
{
    this.Close();
}

```

اگر خواستید موقع بالا آمدن فرم جدول گرید پر شود در رویداد load فرم این کدها را بنویسید:

```

private void studAtr_Load(object sender, EventArgs e)
{
    try
    {
        database2 dbb2 = new database2();
        DataTable DTT2 = dbb2.select("select * from stud");
        dataGridView1.DataSource = DTT2;
        dataGridView1.Columns["studnumber"].HeaderText = " شماره";
        dataGridView1.Columns["name"].HeaderText = " دانشجو نام";
        dataGridView1.Columns["family"].HeaderText = " خانوادگی نام";
        dataGridView1.Columns["term"].HeaderText = " ورودی ترم";
        radioButton1.Checked = true;
    }
    catch { }
}
}

```

اگر خواستید با کلیک موس بر روی سطرهای جدول گرید مشخصات آن بر تکست باکس نمایش داده شود کد

زیر را در رویداد CellMouseClicked بنویسید:

```

private void dataGridView1_CellMouseClicked(object sender,
DataGridViewCellMouseEventArgs e)
{
    try

```

```

        {
            textBox4.Text =
dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();
            textBox1.Text =
dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();
            textBox2.Text =
dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();
            numericUpDown1.Value =
Convert.ToInt32(dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString());
        }
        catch { };
    }
}

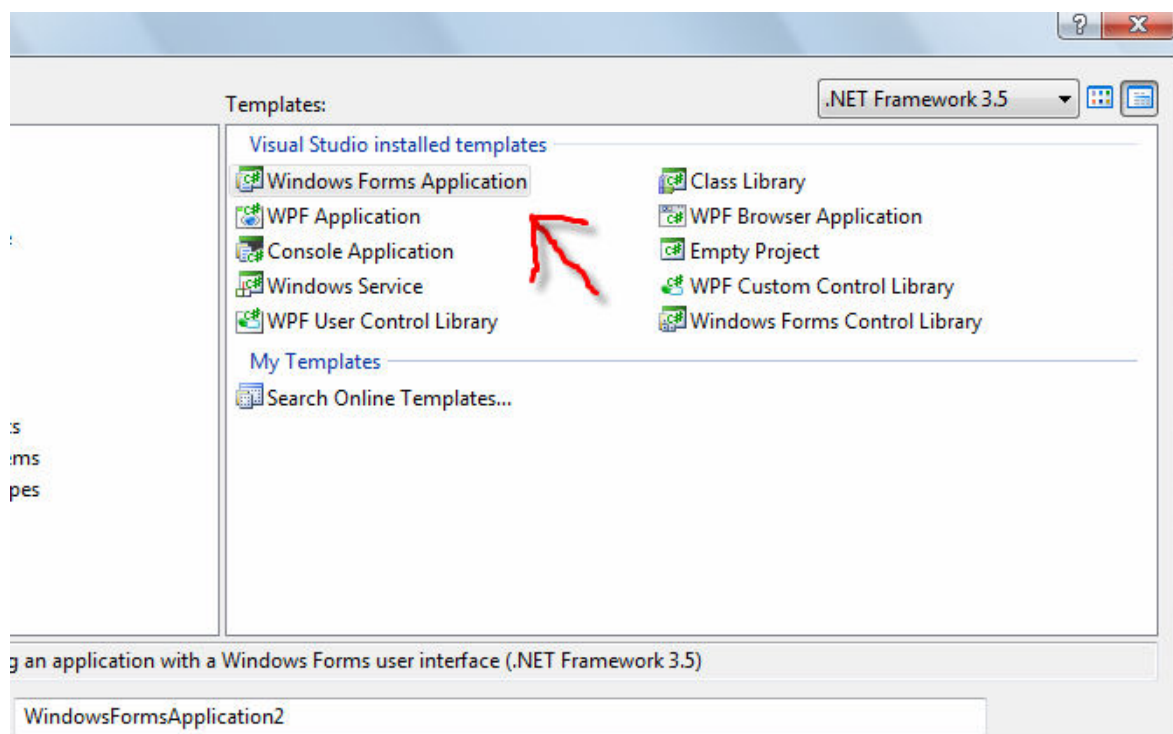
```

حال برنامه را اجرا کنید و نتیجه کار را مشاهده کنید.

### ۳-۶ نحوه گزارش گیری

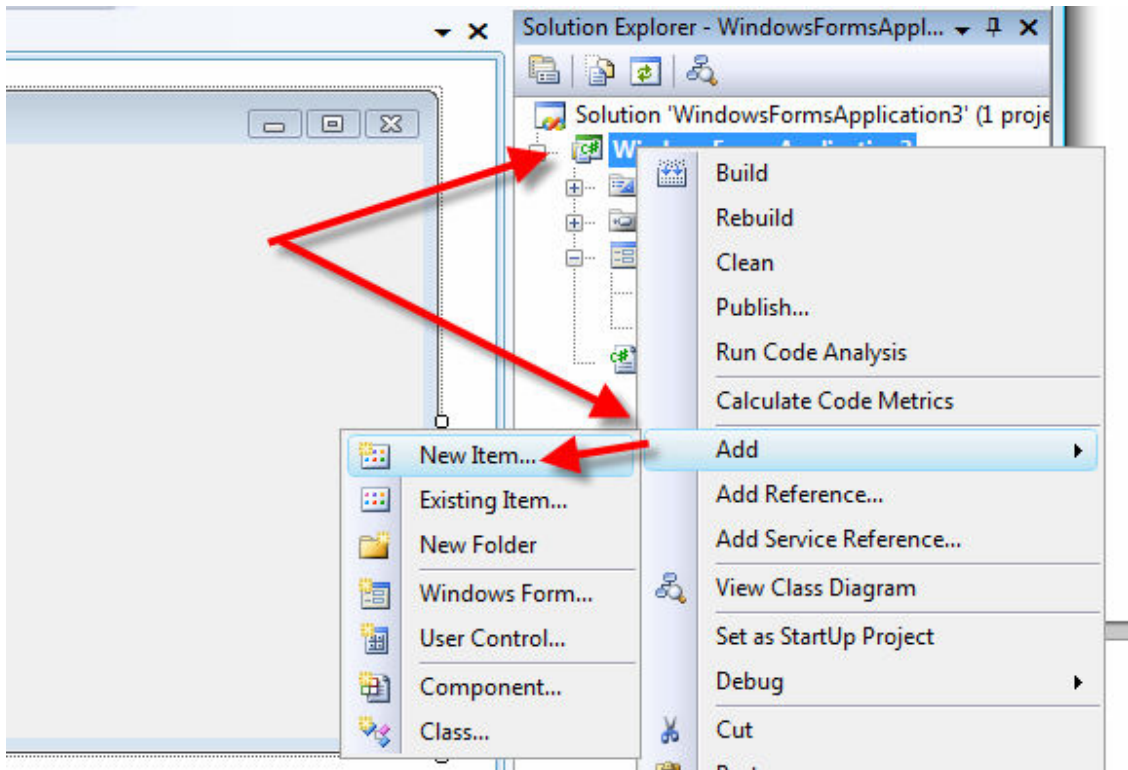
گزارش گیری با استفاده از ابزارهای مهیا کننده ویژوال سی شارپ ۲۰۰۸

برای اینکار یک برنامه ویندوزی ایجاد می کنیم مطابق شکل ۳-۸



شکل ۳-۸ نحوه ایجاد برنامه جدید

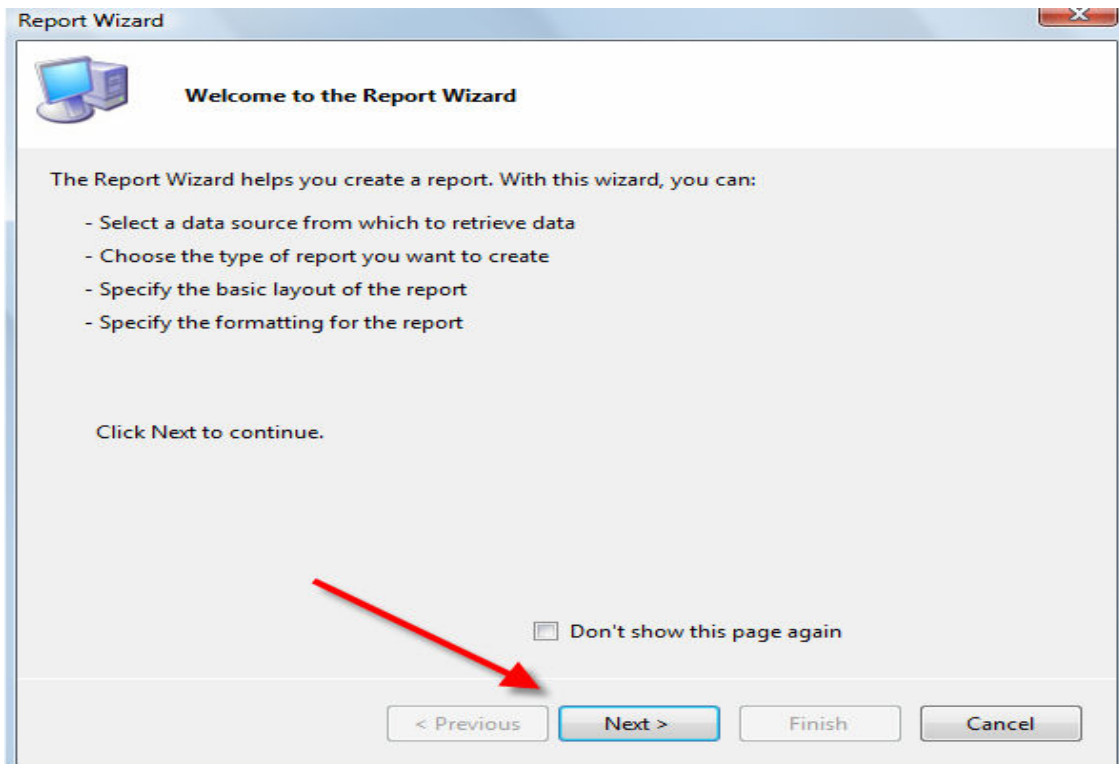
بعد در پنجره Solution مطابق شکل ۳-۹ کلیک نمایید



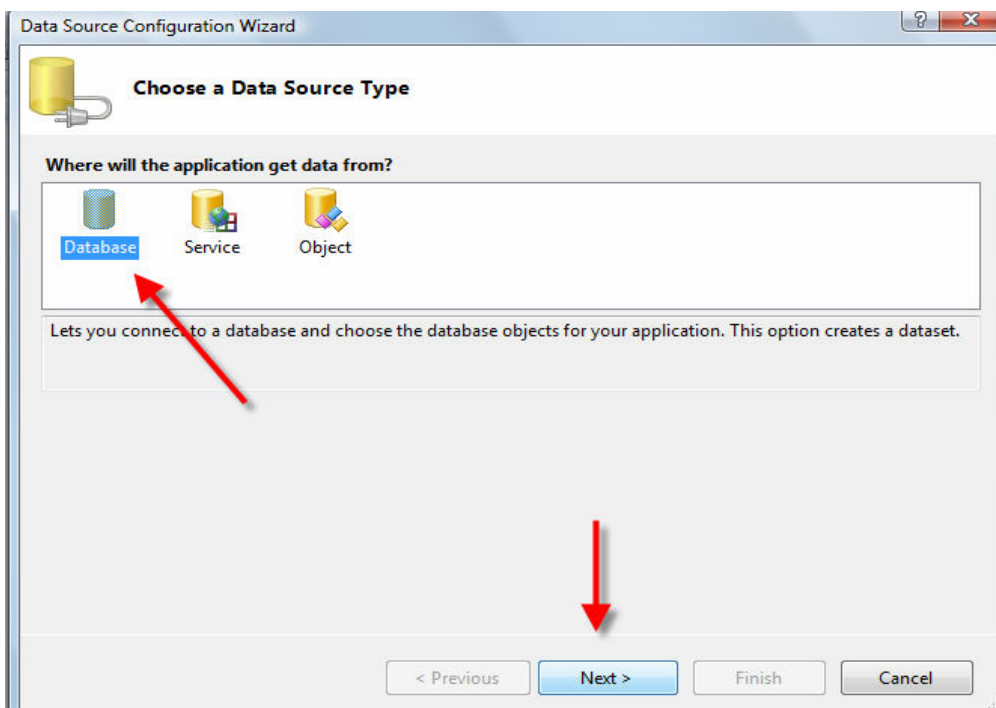
شکل ۳-۹

بعد در پنجره add new item بر روی report wizard کلیک کنید..

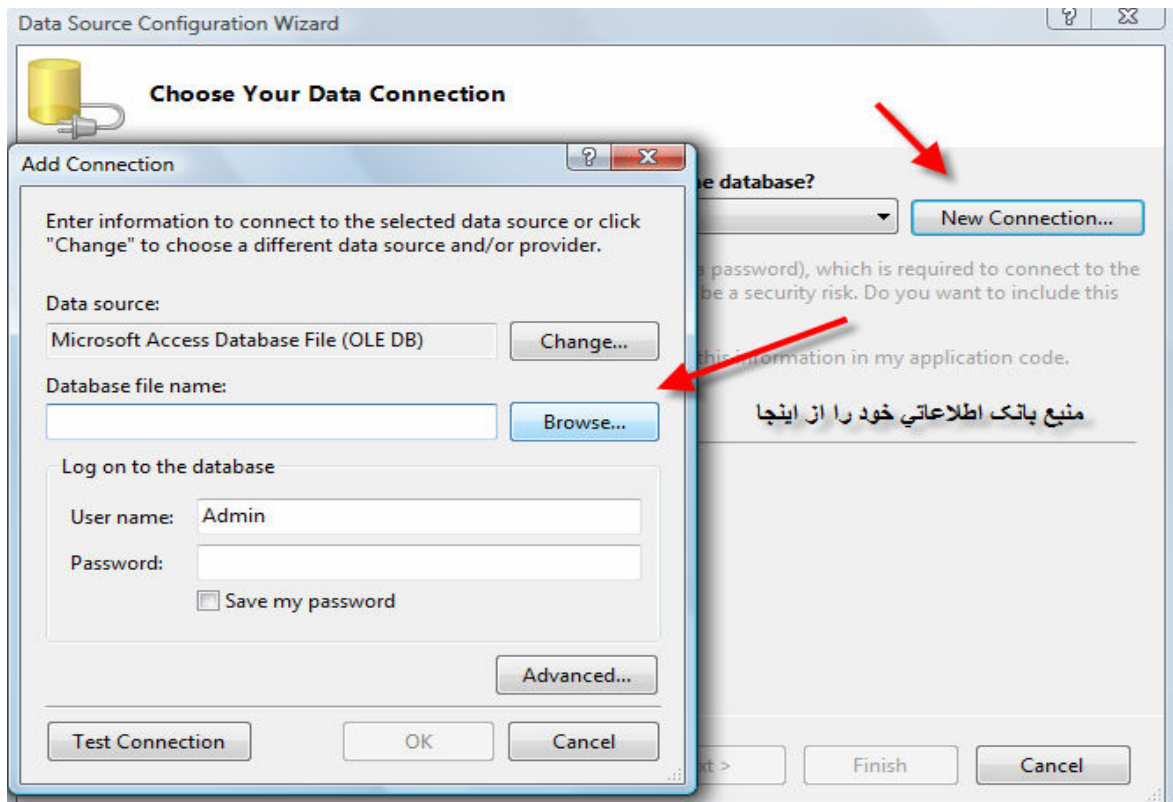
بعد روند زیر را ادامه دهید..درضمن قبلا باید یک بانک اطلاعاتی را توسط یکی از نرم افزارهای مورد نیاز انجام دهید من اینجا از access استفاده کردم .



شکل ۱۰-۳ در این صفحه بر روی دکمه مشخص شده کلیک کنید

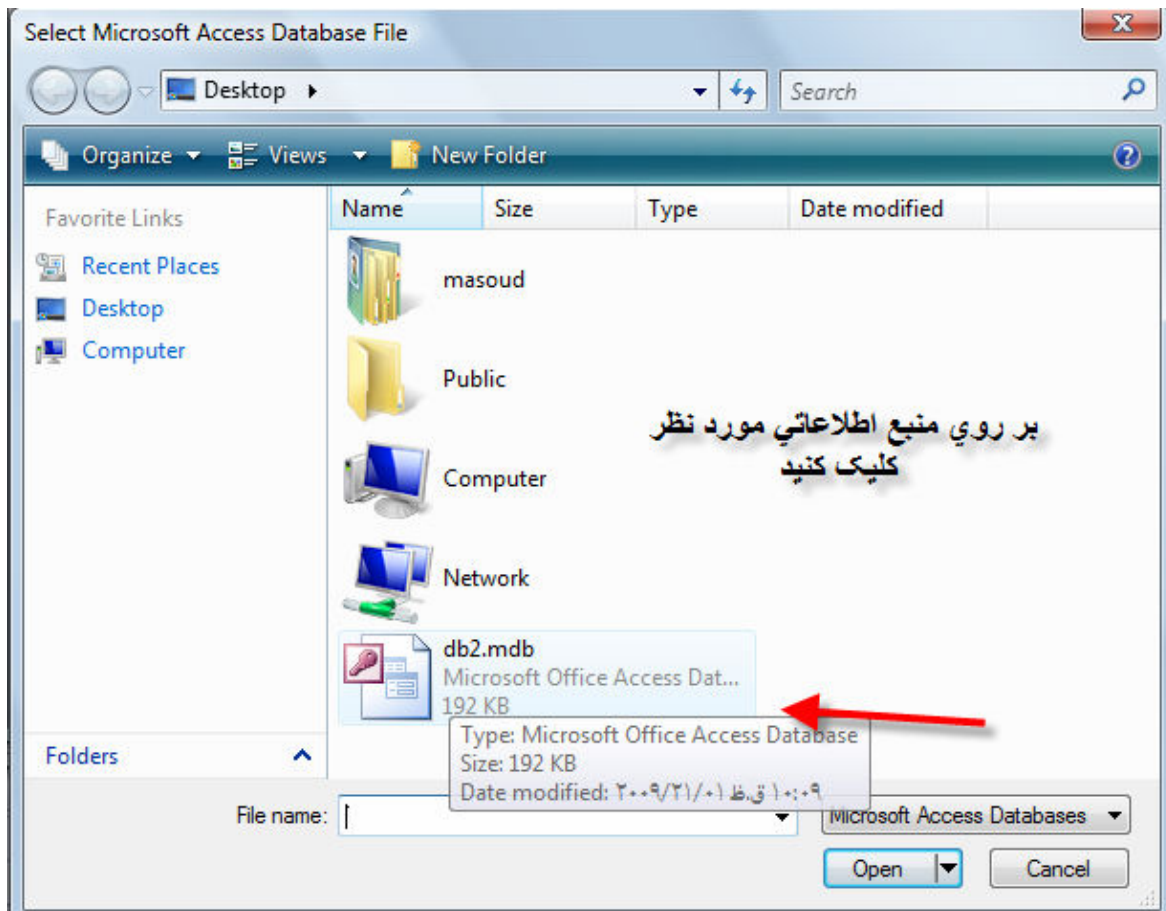


شکل ۱۰-۳



شکل ۳-۱۱





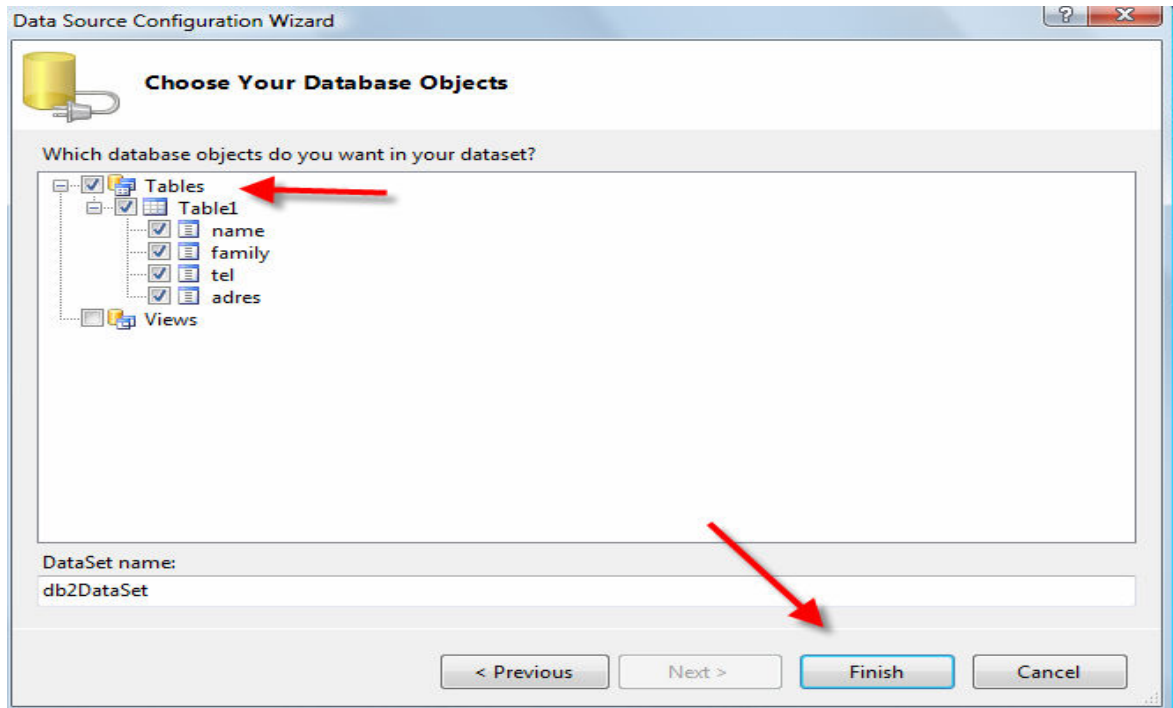
شکل ۳-۱۲

بعد در پنجره add connection بر روی OK کلیک کنید.. بعد دکمه next را بزنید..

بعد از شما پرسیده می شود که آیا یک نسخه از این فایل را بر روی پروژه شما کپی کند که پاسخ Yes را بزنید.

بعد باز هم بر روی دکمه next کلیک کنید تا پنجره زیر ظاهر

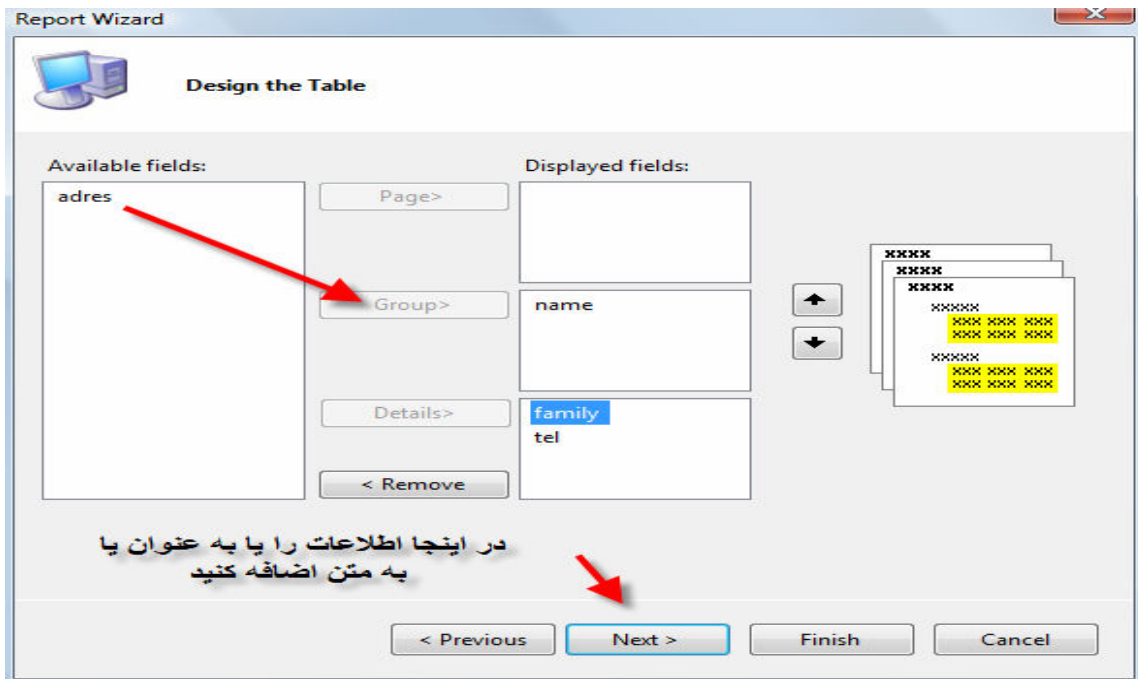
شود



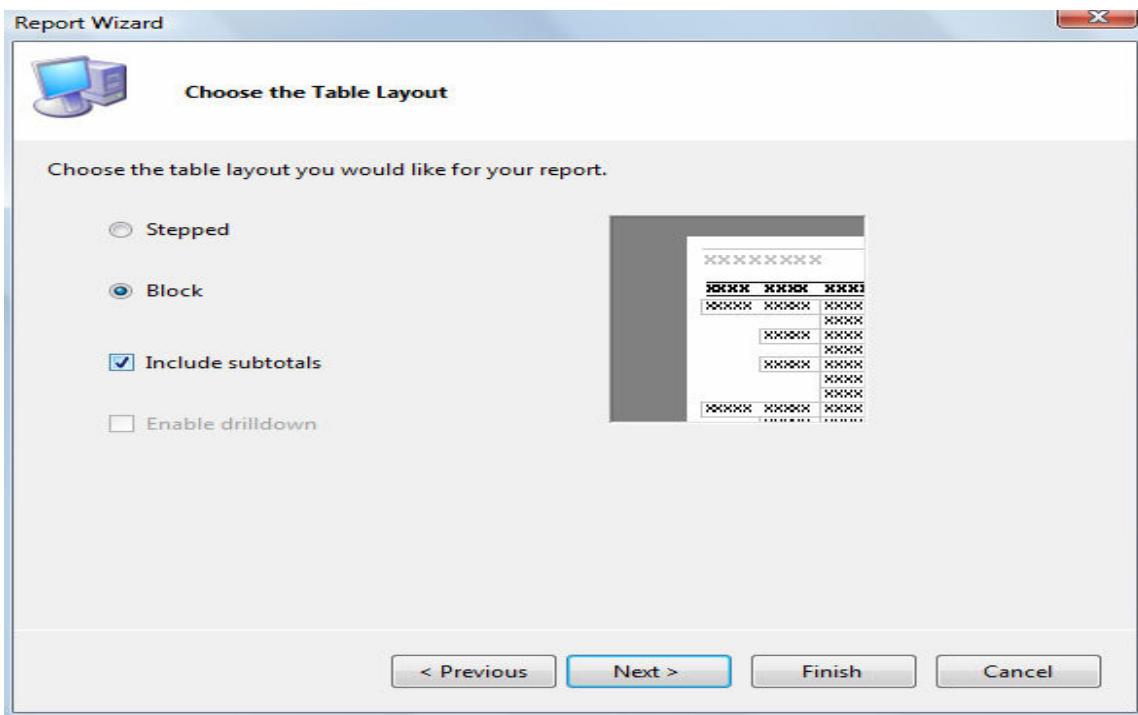
شکل ۳-۱۳

بعد جدول هایی را که میخواهید انتخاب کنید بعد روی finish کلیک کنید.





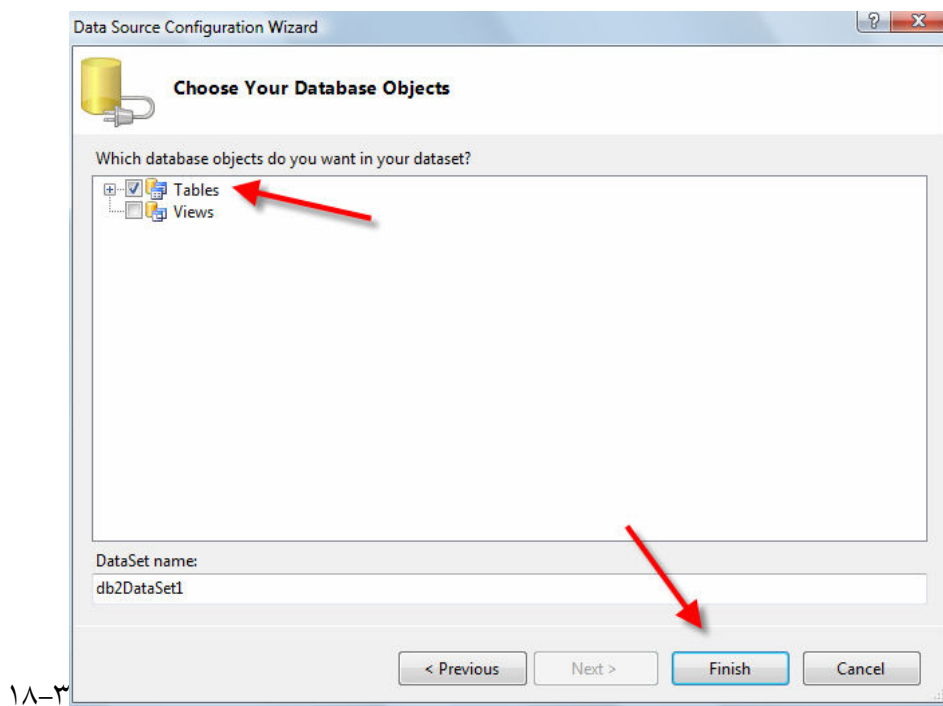
۱۶-۳



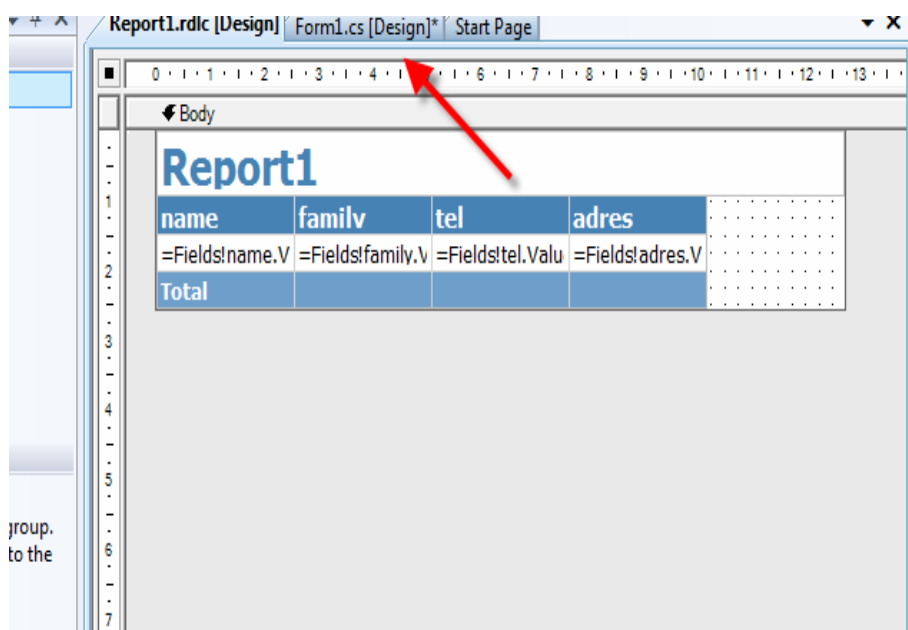
۱۷-۳

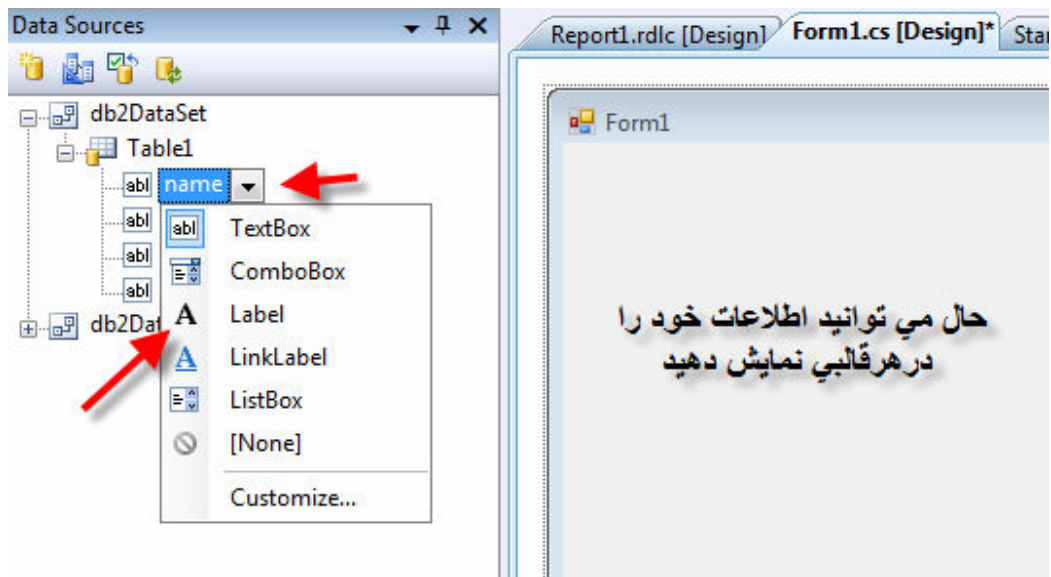
بعد رنگ مورد نظر خود را در صفحه بعدی انتخاب کنید و بر روی **finish** کلیک کنید.

بعد بازم مطابق شکل ۱۸-۳ عمل کنید..

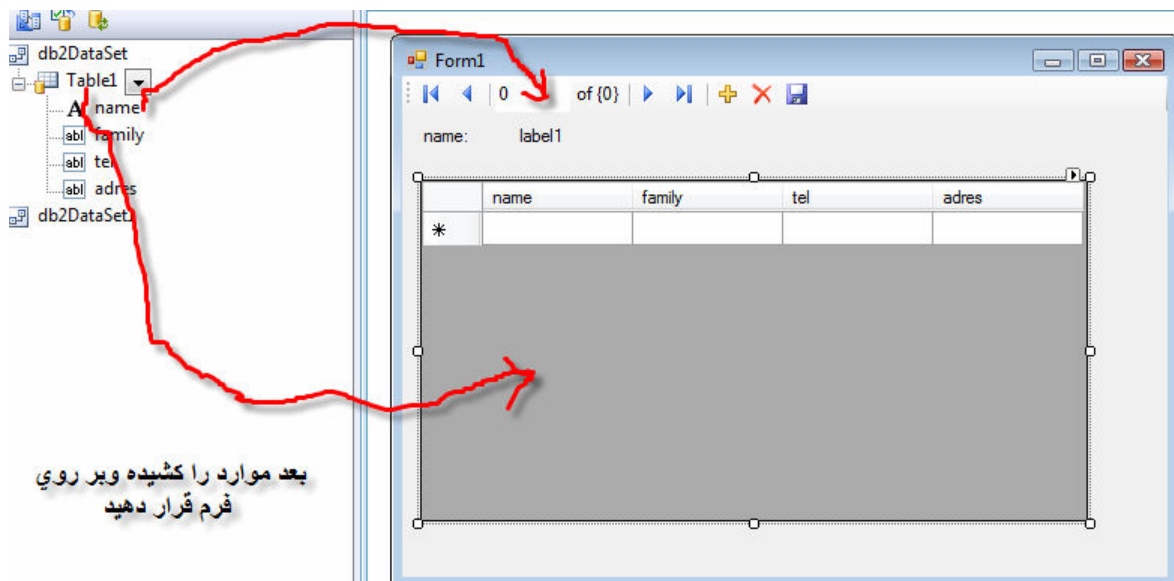


بعد بر روی فرم برنامه خود کلیک کنید مطابق شکل ۱۹-۳





۱۹-۳



۲۰-۳

حال برنامه را اجرا کنید ...

Form1

3 of 6

name: 0

Move next

	name	family	tel	adres
	nader	b	3453	Add1
	javad	c	2223	Add2
▶	0	p	3333	اردبیل
	ابی	رسولی	2342	98322
	als	شسی	2121	01111
	aq	Q	34534534	tehran
*				

۲۱-۳

حال اطلاعات شکل ۲۲-۳ در Access را بر روی فرم مشاهده می کنید می توانید از ابزار های بالا برای اضافه کردن و یا کارهای دیگر استفاده کنید حتی می توانید بر روی جدول نیز عملیات دلخواه را انجام دهید..





کنیم برای مثال در کد زیر [1] آرایه ای از اعداد صحیح تعریف می کنیم و با استفاده از `linq` اعداد زوج آن را انتخاب می کنیم (`select`)

```
private void button1_Click(object sender, EventArgs e)
{
    int[] numbers = new int[7] { 0, 1, 2, 3, 4, 5, 6 };

    var numQuery =
        from num in numbers
        where (num % 2) == 0
        select num;

    foreach (int num in numQuery)
    {
        listBox1.Items.Add(num);
    }
}
```

نکته : `var` نیز مانند یک متغیر عمل می کند که همه نوع را می تواند در خود ذخیره ولی موقعی که مقدار دهی می شود نوع آن تغییر می کند برای مثال در کد زیر وقتی `var` را از نوع `int` مقدار دهی کنیم نوع آن به `int` تغییر می یابد و وقتی آن را با رشته مقدار دهی کنیم نوع آن به `string` تغییر می یابد در مثال بالا نیز نوع آن به آرایه ای از `int` تغییر پیدا کرده است .

```
var i = 9;
var j = "str";
```

مثال : در این مثال از شی های کلاس عمل `select` را انجام می دهیم.

```
private void button1_Click(object sender, EventArgs e)
{
    User[] users = new User[] {
        new User{UserName="Ahmad", Age=20},
        new User{UserName="Maryam", Age=17},
        new User{UserName="Ali", Age=29},
        new User{UserName="Hooman", Age=33},
        new User{UserName="Sara", Age=22},};

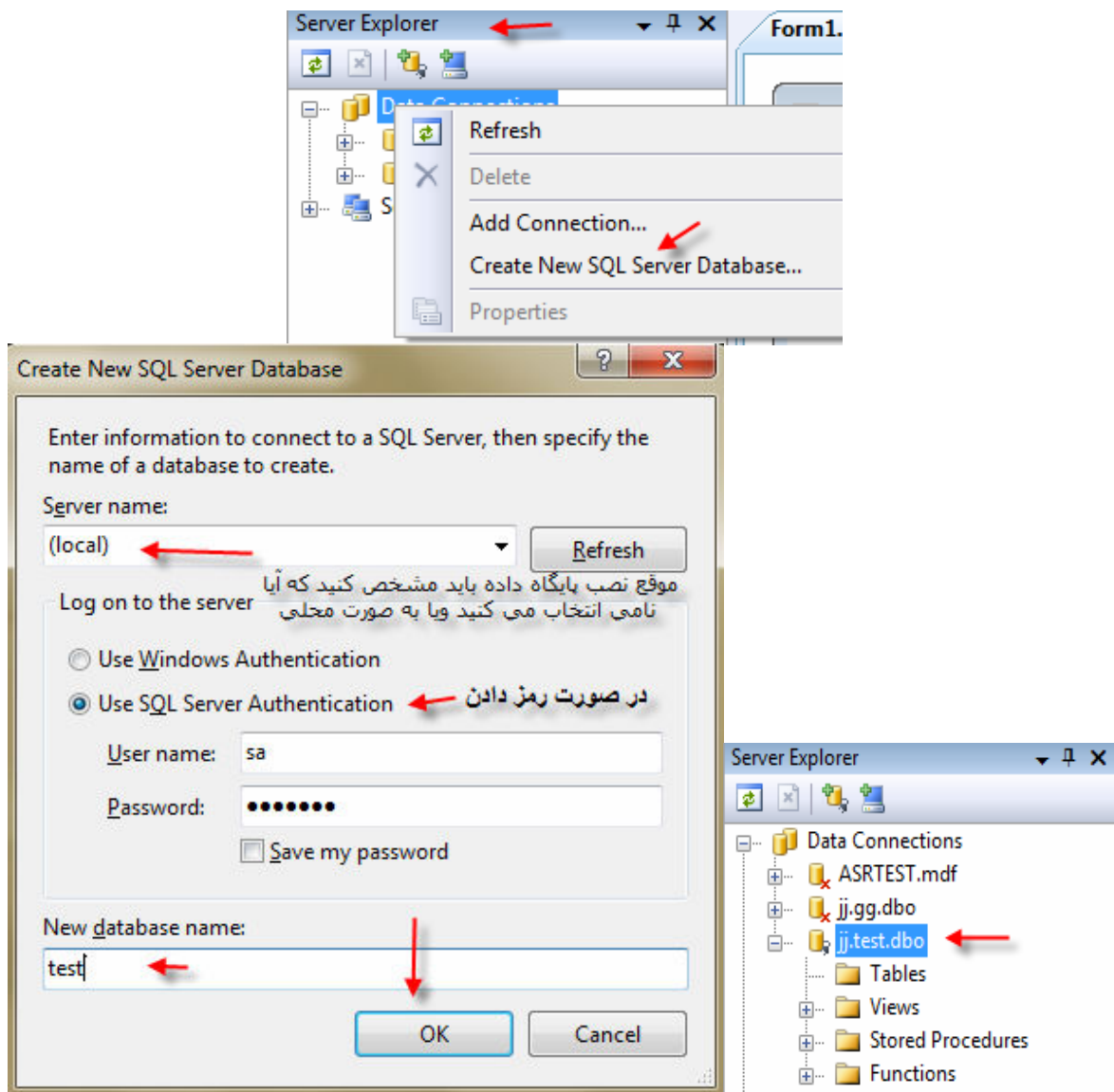
    var mySelect = from user in users where user.Age > 20 && user.Age < 30 select user.UserName;

    foreach (string uname in mySelect)
        listBox1.Items.Add(uname);
}

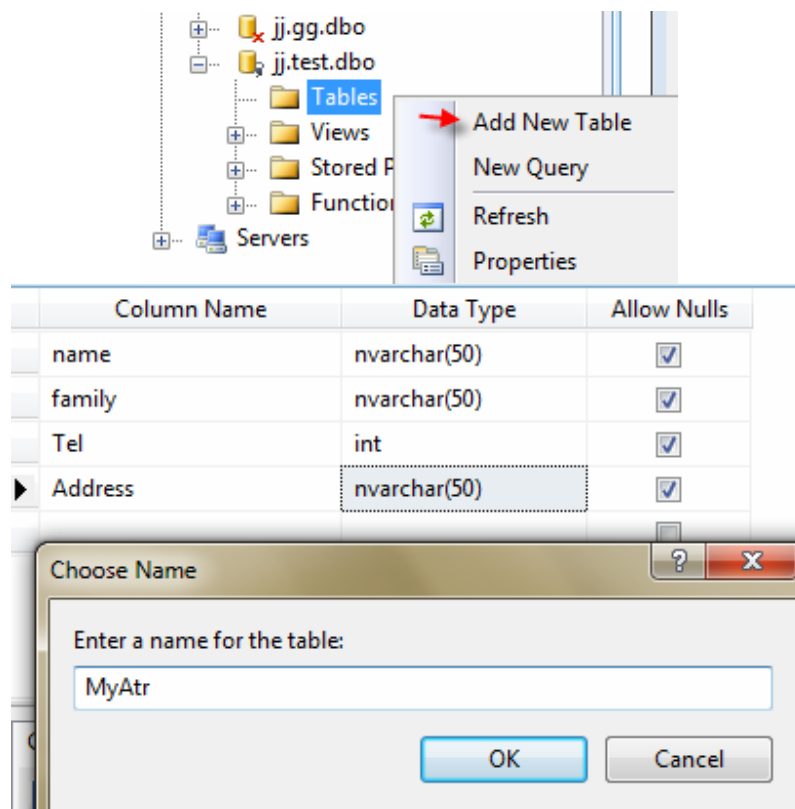
public class User
{
```



قبل از انجام اینکار باید اول پایگاه داده مورد نظر را در sql ایجاد کنیم برای اینکار کافست یا به طور مستقیم به پایگاه داده sql مراجعه کنید و جداول خود را ایجاد کنید یا از طریق سی شارپ به طور غیر مستقیم در server sql ایجاد کنید برای اینکار از پنجره Server Explorer بر روی گزینه Create New SQL Server Database... کلیک کنید و نام سرور خود را انتخاب کنید و اگر رمزی به پایگاه داده دادید آن را نیز مشخص کنید و بعد یک نام برای دیتابیس خود انتخاب کنید.



بعد جدولی با فیلدهای زیر (دلخواه) تعیین کنید و با نام MyAttr ذخیره کنید.



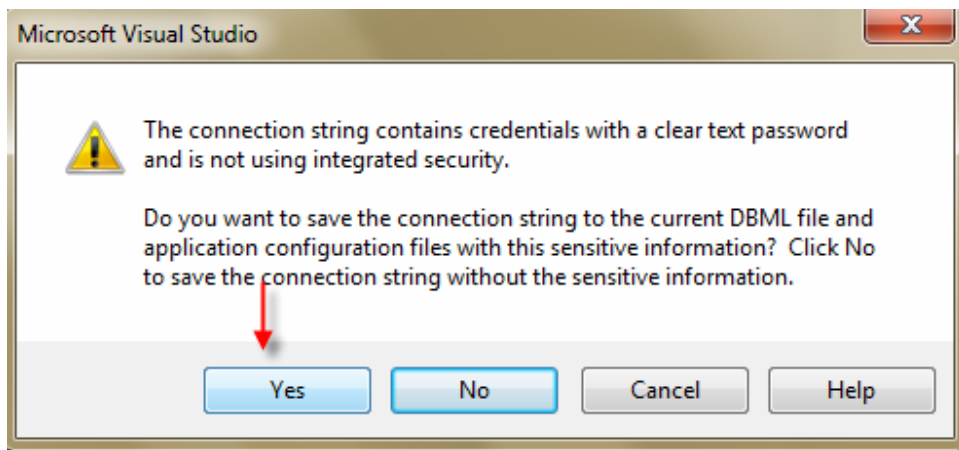
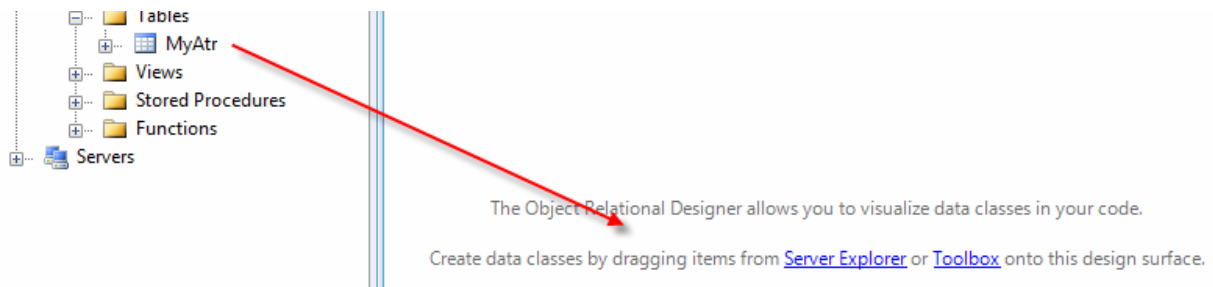
قبل از ذخیره برنامه یک کلید برای جدول انتخاب کنید تا با استفاده از کلید بتوانیم به سطرهایی از جدول احاطه داشته باشیم مثل پاک کردن و یا به روز رسانی بهتر. نام خانوادگی را به عنوان کلید انتخاب کنیم چون خاصیت کلید منحصر به فرد بودن آن است (نباید تکراری باشد) و ما فرض می کنیم نام خانوادگی منحصر به فرد است.

	Column Name	Data Type	Allow Nulls
	name	nvarchar(50)	<input checked="" type="checkbox"/>
	family	nvarchar(50)	<input type="checkbox"/>
	Tel	int	<input checked="" type="checkbox"/>
	Address	nvarchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

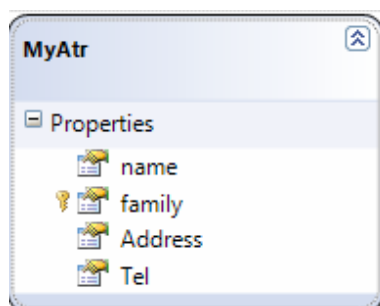
بعد از قسمت Add new item یک linq to sql با نام MyLinq به برنامه اضافه کنید.



بعدانجام اینکار جدول را کلیک کنید و بکشید و بر روی فرم MyLinq.dbml قرار دهید:

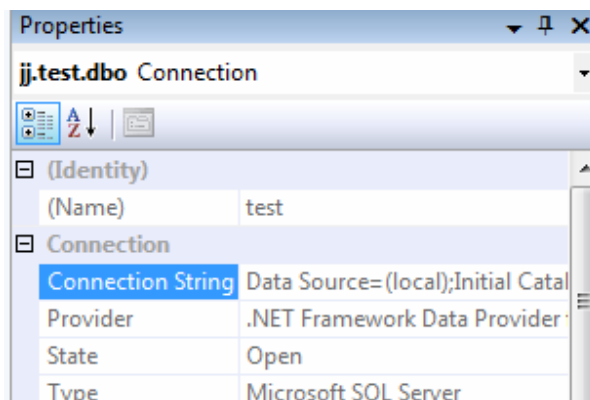


بعد فیلدهای جدول بر روی صفحه نمایش داده می شود



نکته : اطمینان داشته باشید کلید انتخاب شده باشد وگرنه برنامه اجرا نمی شود.

حال برگردید به فرم اصلی تا کدهای مورد نیاز را بنویسیم. بر روی دکمه اضافه کردن کلیک کنید تا شروع به نوشتن کدهایمان بکنیم. در این قسمت می توانیم به راحتی یک شی از کلاس `MyLinqDataContext` ایجاد کنیم و عملیات اضافه کردن را به سادگی انجام بدهیم. این کلاس نیاز به یک `Connection string` دارد که می توانید آن را از `properties` دیتابیس به دست بیاورید:



بعد یک کلاس از نام جدولمان ایجاد کنید و فیلدهای آن را به سادگی مقدار دهید کنید:

```
string str = "Data Source=(local);Initial Catalog=test;User ID=sa;Password=666666;Pooling=False";
private void button1_Click(object sender, EventArgs e)
{
    MyLinqDataContext _MS = new MyLinqDataContext(str);
    MyAtr _MYdata = new MyAtr();
    _MYdata.name = textBox1.Text;
    _MYdata.family = textBox2.Text;
    _MYdata.Tel = int.Parse(textBox3.Text);
    _MYdata.Address = textBox4.Text;
    _MS.MyAtrs.InsertOnSubmit(_MYdata);
    _MS.SubmitChanges();
    MessageBox.Show("insert");
}
```

روش کار به این صورت است که فیلدهای کلاس `MyAtr` را مقدار دهی می کنیم بعد توسط متد `SubmitChanges()` از کلاس `DataContext` این اطلاعات را ذخیره می کنیم.

بعد بر روی نام جدول کلیک راست کرده و show Table data و اطلاعات جدول نمایش داده شود.

	name	family	Address	Tel
▶	o	k	o o	1
	p	p	pdsf	123
*	NULL	NULL	NULL	NULL

کدهای مربوط به پاک کردن بر اساس نام خانوادگی:

```
private void button2_Click(object sender, EventArgs e)
{
    MyLinqDataContext _MS = new MyLinqDataContext(str);

    var deleteOrderDetails =
from details in _MS.MyAtrs
where details.family==textBox2.Text
select details;

    foreach (var detail in deleteOrderDetails)
    {
        _MS.MyAtrs.DeleteOnSubmit(detail);
    }

    _MS.SubmitChanges();
    MessageBox.Show("Delete");
}
```

کدهای مربوط به دکمه به روز رسانی:

```
private void button3_Click(object sender, EventArgs e)
{
    MyLinqDataContext _MS = new MyLinqDataContext(str);
    var tb = _MS.MyAtrs.Where(p => p.family.Equals(textBox2.Text));
    foreach (var roe in tb)
    {
        roe.name = textBox1.Text;
        roe.Tel = int.Parse(textBox3.Text);
        roe.Address = textBox4.Text;
    }
    _MS.SubmitChanges();
    MessageBox.Show("Update");
}
```



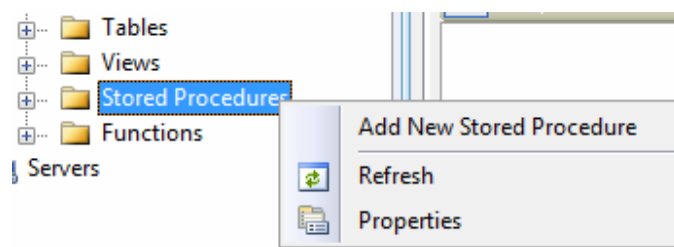


مثال : برنامه ای بنویسید که مشخصات فردی افراد را با استفاده از پروسیجر (PROCEDURE) ها در پایگاه داده sql server ذخیره کند.

برای انجام اینکار یک دیتابیس با نام دلخواه ایجاد کنید و فیلدهای جدول آن را نیز به دلخواه ایجاد کنید. روند ایجاد این مراحل را در مثال قبلی توضیح دادم

Column Name	Data Type	Allow Nulls
Mname	nvarchar(50)	<input checked="" type="checkbox"/>
Mfamily	nvarchar(50)	<input checked="" type="checkbox"/>
Tel	int	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

بعد از ایجاد جدول نوبت به ایجاد پروسیجر ها می رسد برای اینکار بر روی Stored Procedures کلیک راست کرده و یک پروسیجر جدید ایجاد کنید مطابق شکل زیر:



```
CREATE PROCEDURE dbo.StoredProcedure1
/*
(
@parameter1 int = 5,
@parameter2 datatype OUTPUT
)
*/
AS
/* SET NOCOUNT ON */
RETURN
```

بعد دستورات لازم را درون پروسیجر ایجاد کنید مثل دستور insert

```
ALTER PROCEDURE dbo.StoredProcedure3
@Mname nvarchar(50),
@Mfamily nvarchar(50),
@Tel int
```

```
AS
insert into Table1 values (@Mname,@Mfamily,@Tel)
RETURN
```

در کد بالا مقادیری که از سی شارپ می فرستیم درون پارامترها قرار می گیرد مثل پارامتر @Mname بعد این پارامترها به جدول اضافه می شود . در محیط سی شارپ نیز فقط لازم است نام پروسیجر را همراه با پارامترهای ارسالی بنویسیم مثل کد زیر:

```
private void button1_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection();
    SqlCommand cmd = new SqlCommand();

    con.ConnectionString = "Data Source=(local);Initial
Catalog=TestMN;User ID=sa;Password=666666;Pooling=False";
    con.Open();
    cmd.Connection = con;
    cmd.CommandText = "StoredProcedure3";
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@Mname", SqlDbType.NVarChar).Value =
textBox1.Text;
    cmd.Parameters.Add("@Mfamily", SqlDbType.NVarChar).Value =
textBox2.Text;
    cmd.Parameters.Add("@Tel", SqlDbType.Int).Value =
int.Parse(textBox3.Text);
    cmd.ExecuteNonQuery();
    con.Close();
}
```

در کد بالا به جای نوشتن مستقیم دستورات sql در CommandText فقط نام پروسیجر را می نویسم و CommandType برابر با CommandType.StoredProcedure قرار می دهیم و پارامترهای مورد نیاز را با استفاده از Parameters.Add اضافه می کنیم . معادل دستور بالا را به شکل کد زیر نیز می توانیم بنویسیم یعنی بدون استفاده از پروسیجرها:

```
private void button1_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection();
    SqlCommand cmd = new SqlCommand();

    con.ConnectionString = "Data Source=(local);Initial
Catalog=TestMN;User ID=sa;Password=666666;Pooling=False";
    con.Open();
    cmd.Connection = con;
    cmd.CommandText = "insert into Table1
values (@Mname,@Mfamily,@Tel) ";
}
```

```

        cmd.Parameters.Add("@Mname", SqlDbType.NVarChar).Value =
textBox1.Text;
        cmd.Parameters.Add("@Mfamily", SqlDbType.NVarChar).Value =
textBox2.Text;
        cmd.Parameters.Add("@Tel", SqlDbType.Int).Value =
int.Parse(textBox3.Text);
        cmd.ExecuteNonQuery();
        con.Close();
    }

```

برای پر کردن جدول گرید ویو باید یک پروسیجر دیگر ایجاد کنیم که عمل **select** را برای ما انجام دهد :

```

ALTER PROCEDURE dbo.StoredProcedure1
AS
    select * from Table1
RETURN

```

بعد کد زیر را در سی شارپ می نویسیم:

```

private void button2_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection();
    SqlCommand cmd = new SqlCommand();

    DataTable _DA = new DataTable();
    con.ConnectionString = "Data Source=(local);Initial
Catalog=TestMN;User ID=sa;Password=666666;Pooling=False";
    con.Open();
    cmd.Connection = con;
    cmd.CommandText = "StoredProcedure1";
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataAdapter _SDA = new SqlDataAdapter(cmd);
    _SDA.Fill(_DA);
    cmd.ExecuteNonQuery();
    dataGridView1.DataSource = _DA;
    con.Close();
}

```

حال می خواهیم مقادیری را از جدولمان با استفاده از پروسیجرها بخوانیم و برگشت بدیم . در کد زیر ابتدا یک مقدار به پروسیجر می فرستیم و پروسیجر شرطی را بر روی آن انجام می دهد و اگر آن شرط برقرار بود مقادیری را برای ما بر می گرداند:

```

ALTER PROCEDURE dbo.StoredProcedure2
@Mname nvarchar(50) output,
    @Mfamily nvarchar(50) output,
    @Tel int output,
    @Search nvarchar(50)
AS

```



فصل چهارم :

# ASP . NET

در این بخش می خواهیم درباره صفحات اینترنت بحث کنیم اینکه چگونه می توانیم یک سایت با استفاده از تکنولوژی asp در سی شارپ طراحی کنیم ولی قبل ازاینکه به این بحث پردازیم ابتدا توضیحاتی در پیرامون این موضوع ارایه می دهیم .

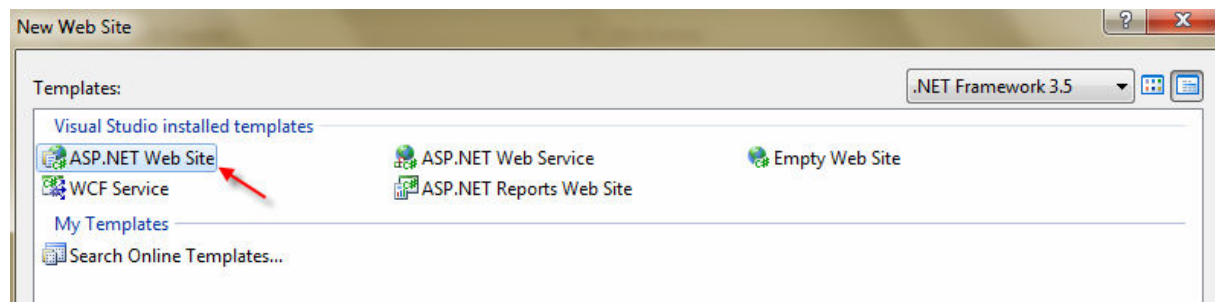
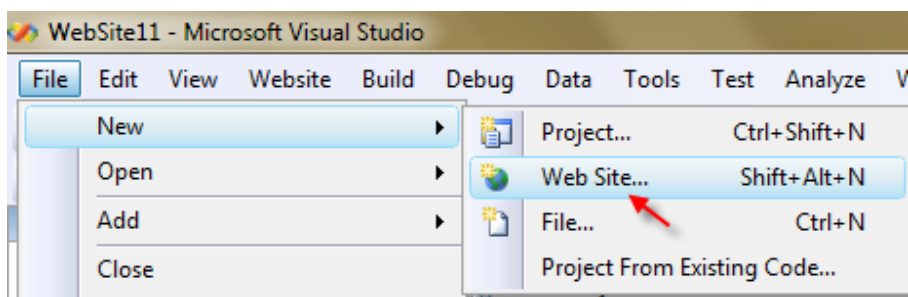
حتما تا حالا به اینترنت برای انجام کارهایتان مراجعه کردید مثل ثبت نام و خرید اینترنتی و یا پرداخت پول یا چیزهای دیگر و شاید برای شما سوالهای بسیاری پیش آمده مثل نحوه ساخت این سایتها و



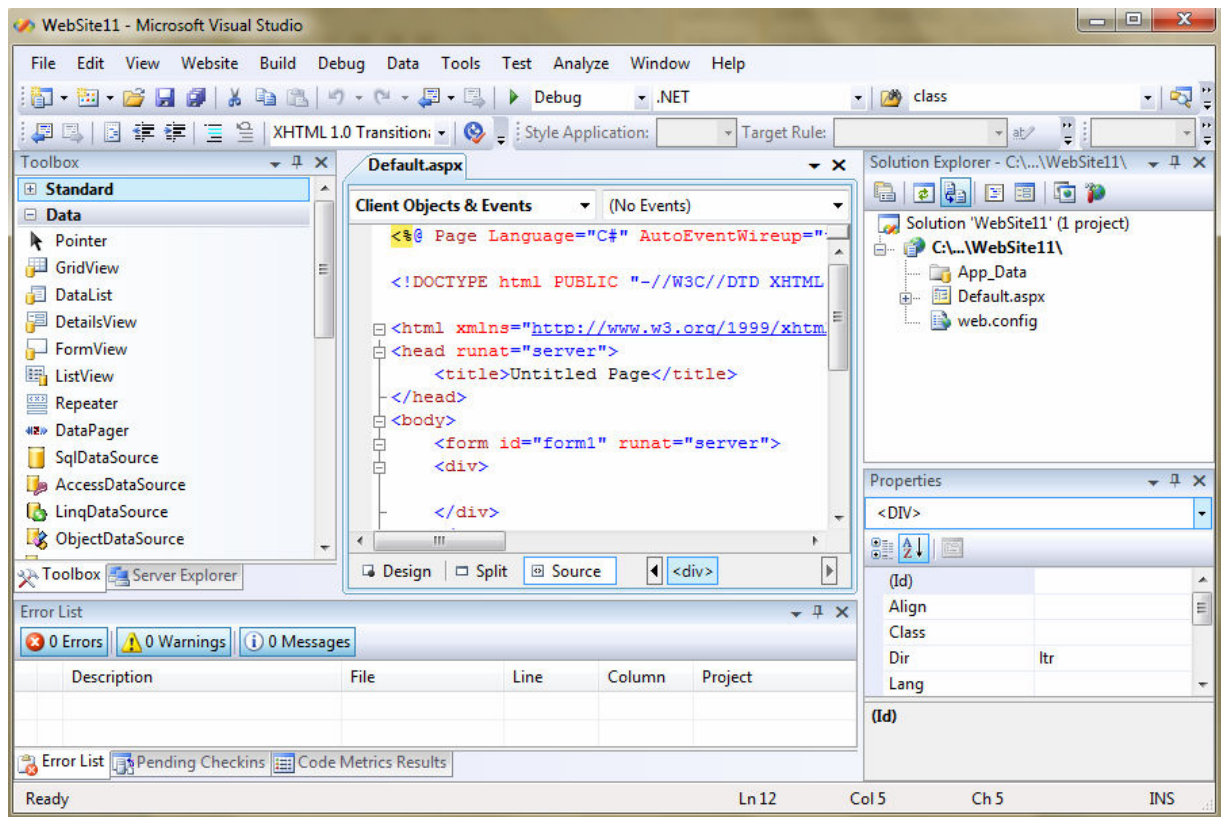
نمایش آنها و حتی شکل ظاهری آنها . دراین فصل می خواهیم به این سوالات پاسخ دهیم .نحوه ساخت این صفحات شبیه ساخت فرمهایی است که در فصلهای قبل با آنها آشنا شدیم ولی تفاوت های اساسی بسیاری در اینجا به چشم می خورد که به آنها خواهیم پرداخت .صفحات از چندین لایه تشکیل شده اند یک لایه مربوط می شود به لایه طراحی صفحات که بیشتر دراین لایه با زبان های مختلف از جمله

Html, ajax, asp سرو کار داریم طراحی این لایه از اهمیت خاصی برخوردار است چون اگر صفحاتی را به زیباترین (محیط آسان و دلنشین) شکل طراحی کنیم ممکن است کاربران زیادی را به خود جلب کندونیز باید صفحات را طوری طراحی کنیم که دارای سرعت بیشتری باشد. در این لایه نوشتن کدهای برنامه سخت است (مثلا وصل شدن به پایگاههای داده) بنابراین محیط دیگری وجود دارد که در آن می توانیم به زبان سی شارپ و یا زبان های دیگر کدهایمان را بنویسیم.

نحوه عملکرد صفحات اینترنتی به کلی با windows application فرق دارد در برنامه های تحت ویندوز وقتی برنامه ای می نوشتیم این برنامه در محیط ویندوز اجرا می شد بدون اینکه نیاز باشد اطلاعاتی را در شبکه بین چندین کامپیوتر رد و بدل شود ولی در صفحات اینترنت وقتی ما در کامپیوترمان یک آدرس مثل [www.yahoo.com](http://www.yahoo.com) می نویسیم در واقع یک درخواست به سرور یاهو می فرستیم برای باز کردن سایت یاهو و در پاسخ به درخواست ما سرور یاهو به ما پاسخ می دهد و صفحات مورد نیاز را در اختیار ما قرار می دهد وقتی ما درخواستی به یک سایت می دهیم در خواست ما با استفاده از پروتکل خاص بسته بندی شده و به سایت مورد نظر فرستاده می شود و برعکس. ولی ما نیازی نداریم به اینکه بدانیم این کارها چگونه انجام می گیرد در این کتاب فقط به نحوه ساخت سایت می پردازیم و کاری به جزئیات کار نداریم. برای انجام اینکار یک پروژه وب سایت با نام دلخواه ایجاد کنید:



حال وارد محیط طراحی سایت می شویم :



در زبان html و دیگر زبان های ساخت صفحات وب همه کدهای بین تگ ها نوشته می شوند مثل کد زیر که ابتدا نام تگ مشخص شده و کدهای بین دو تگ نوشته می شود

```
<body>
////////////////////code
</body>
```

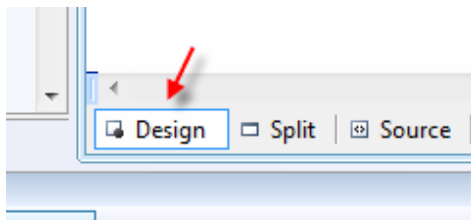
وقتی شما پروژه جدیدی ایجاد کردید کدهایی شبیه این را مشاهده می کنید مثل :

```
<body>
    <form id="form1" runat="server">
    <div>

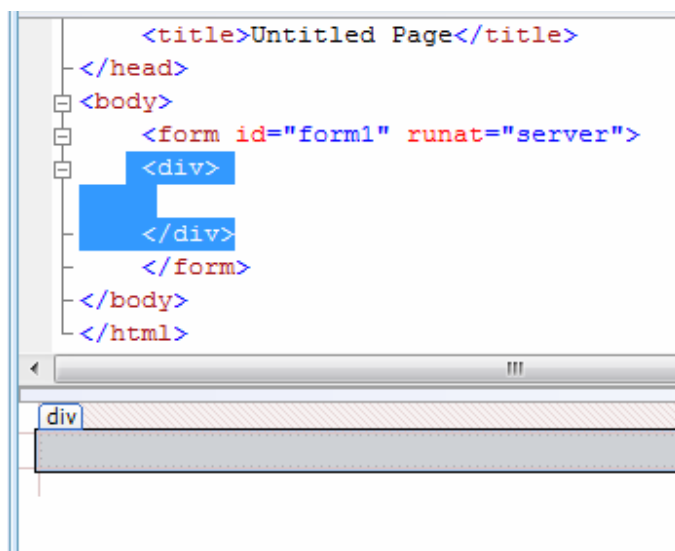
    </div>
    </form>
</body>
```



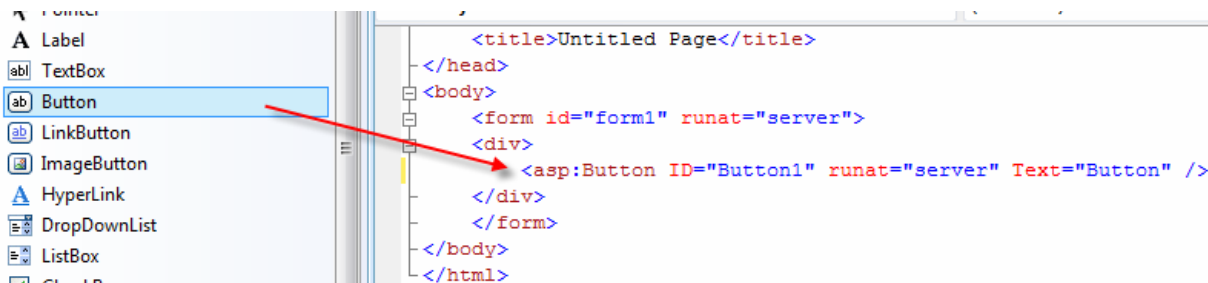
همانطور که ملاحظه کردید همه کد های بین دو تگ ابتدایی و انتهایی نوشته می شود برای مشاهده محیط طراحی می توانید بر روی Design کلیک کنید



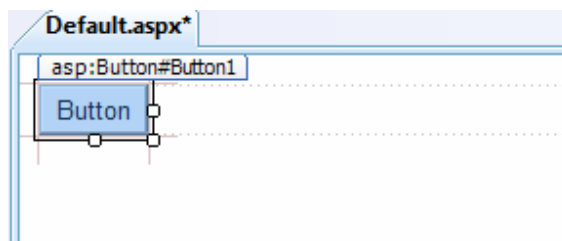
در ابتدا صفحه طراحی خالی است و فقط یک جعبه خالی را مشاهده می کنید که همان ابزار div است



حال در قسمت source بین دو تگ div یک دکمه قرار دهید برای انجام اینکار دکمه را بکشید و بین فضای خالی div قرار دهید. مثل شکل زیر:

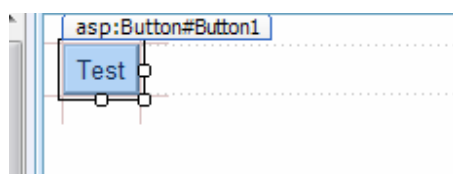


حال به محیط Design برگردید و شکل دکمه را مشاهده کنید: (درای محیط نیز می توانید ابزار را قرار دهید)



بعد برگردید و text آن را تغییر دهید و نتیجه را مشاهده کنید. مثل کد زیر :

```
<div>
  <asp:Button ID="Button1" runat="server" Text="Test" />
</div>
```

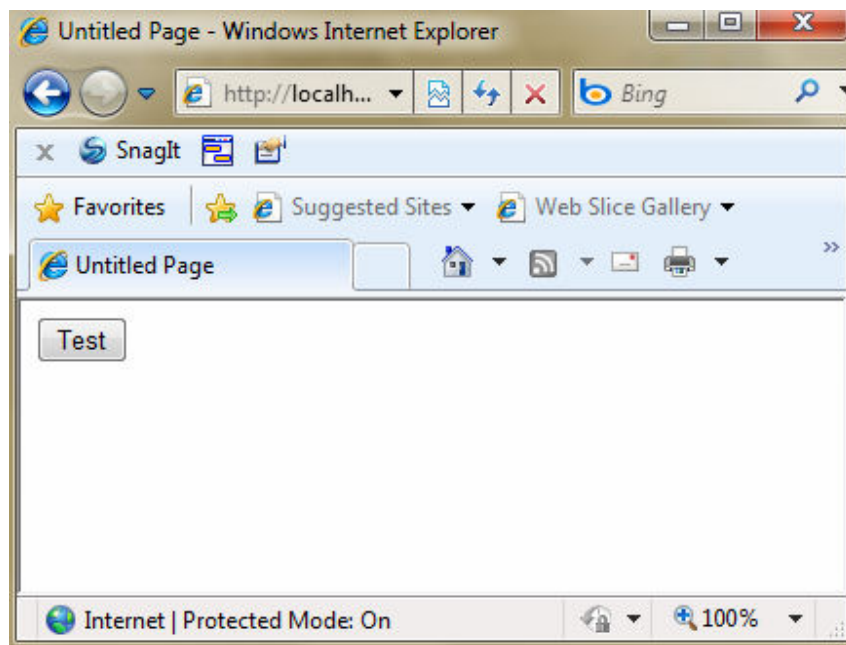


همانطور که مشاهده کردید با تغییر در محیط کد نویسی می توانید شکل برنامه را تغییر دهید حال برنامه را با `ctrl+F5` اجرا کنید و نتیجه را مشاهده کنید:

نکته : کدهایی که با `asp:` نوشته می شوند یعنی از تکنولوژی `asp` استفاده می کنند بنابراین کدهایی مثل `div` به زبان `html` نوشته می شود.

نکته : کدهای صفحات وب توسط `Internet Explorer` یا دیگر برورزها نمایش داده می شود.

نکته : با قرار دادن `runat="server"` می خواهیم کدهای مربوط به کنترل در سمت سرور اجرا شود.



اگر بر روی دکمه کلیک کنید هیچ اتفاقی نمی افتد چون هنوز کدی برای آن ننوشتیم و فقط طراحی انجام دادیم برای نوشتن کد باید از سی شارپ استفاده کنیم .

مثال : یک تکست باکس بعد از کد بالا به شکل زیر قرار دهید:

```
<div>
  <asp:Button ID="Button1" runat="server" Text="Test" />
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</div>
</form>
```



اگر خواستید که تکست باکس در سطر پایین قرار گیرد از Br استفاده کنید مثل کد زیر:

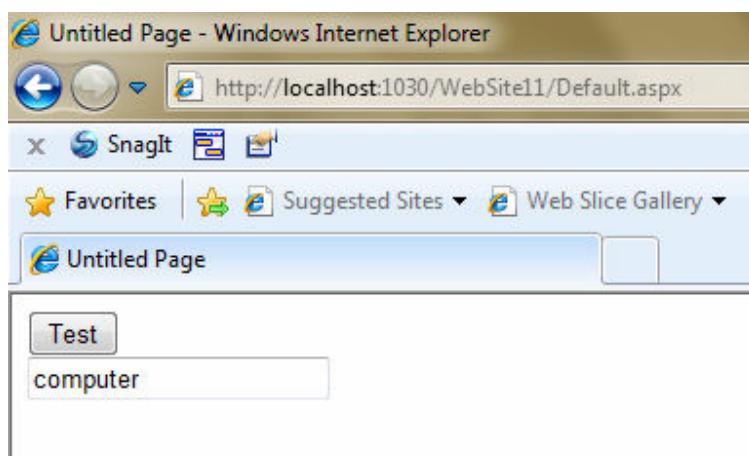
```
<div>
  <asp:Button ID="Button1" runat="server" Text="Test" />
  <br />
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</div>
```



بعد بر روی دکمه دوبار کلیک متوالی کنید تا وارد محیط برنامه نویسی شود و کد زیر را برای رویداد کلیک آن بنویسید با این کار در واقع ما از سرور درخواست می کنیم که متن داخل تکست باکس ما را با متن مشخص شده پر کند.

```
protected void Button1_Click(object sender, EventArgs e)
{
    TextBox1.Text = "computer";
}
```

برنامه را اجرا کنید و نتیجه را مشاهده فرمایید.



اگر خواستید رنگ پشت زمینه دکمه را عوض کنید می توانید `BackColor` مثل کد زیر استفاده کنید:

```
<asp:Button ID="Button1" runat="server" Text="Test"
onclick="Button1_Click" BackColor="Blue" />
```

نکته :

وقتی شما برنامه را اجرا می کنید و دکمه `Test` را می زنید (یا هر عمل دیگری) در زیرصفحه یک شکل به

رنگ سبز تا پایان کار به ترتیب پر میشود. دلیل آن اینست که وقتی شما



دکمه را می فشارید در واقع درخواستی به سرور می فرستید و سرور به درخواست شما پاسخ می دهد مثلا در این برنامه شما درخواست کردید که تکست باکس را برایم با نام کامپیوتر پر کند و سرور این عمل را انجام می دهد و نتیجه را برای شما می فرستد و شما در صفحه ای جدید این درخواست را مشاهده می کنید ( البته چون در اینجا سرور مجازی و برای امتحان برنامه است و در خود سی شارپ قرار دارد این کار به سرعت انجام می گیرد ).

برگردیم به برنامه . اگر خواستید به برنامه توضیحاتی اضافه کنید هم می توانید از label استفاده کنید و هم اینکه به صورت مستقیم متن خود را تایپ کنید:

```
<div>
  <asp:Button ID="Button1" runat="server" Text="Test"
onclick="Button1_Click" BackColor="Blue" />
  <br />
  کامپیوتر
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</div>
```



چون همه این ابزارها درون تگ div قرار گرفته بنابراین اگر ویژگی های آن را تغییر دهید ممکن است تغییراتی در کل ابزارها صورت گیرد برای تغییر ویژگی ها می توانید از style استفاده کنید.

```
<div style="font-family:Tahoma;background-color:Green">
  <asp:Button ID="Button1" runat="server" Text="کنید کلیک"
onclick="Button1_Click" />
  <br />
  کامپیوتر
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</div>
```

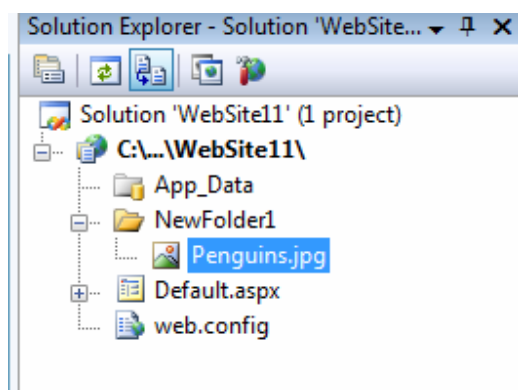
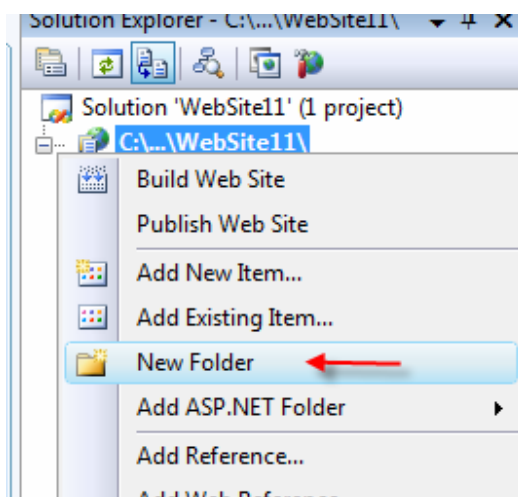
مثلا اگر خواستید همه ابزارها در وسط صفحه قرار گیرند باید خاصیت text-align آن را تغییر دهید به center

```
<div style="font-family:Tahoma;background-color:Green;text-align:center">
  <asp:Button ID="Button1" runat="server" Text="کنید کلیک"
onclick="Button1_Click" />
  <br />
  کامپیوتر
  <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
</div>
```

برای قرار دادن عکس بر روی صفحه باید به این شکل عمل کنید:

ابتدا یک newfolder با نام دلخواه ایجاد کنید و کلیک راست کرده و یک عکس درون این پوشه کپی کنید

:



و یک Image در قسمت کد قرار دهید

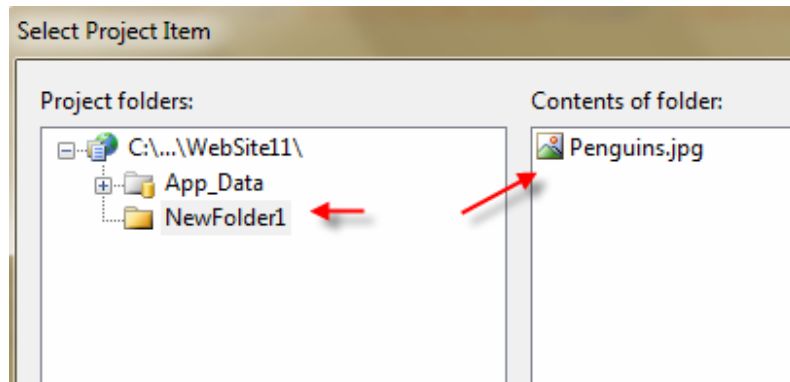
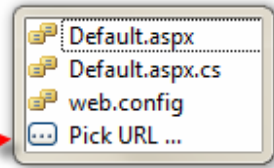
```
<asp:Image ID="Image1" runat="server" />
```

و `ImageUrl` را برابر با عکس قرار دهید برای اینکار کافیسست مثل شکل زیر عمل کنید

```

<br />
<asp:Image ID="Image1" runat="server" ImageUrl= />
</div>
</form>
<p>
&nbsp;  </p>

```



یا اگر مسیر را بلد باشید می توانید آن را مستقیم تایپ کنید:

```

<asp:Image ID="Image1" runat="server"
ImageUrl="~/NewFolder1/Penguins.jpg" />

```

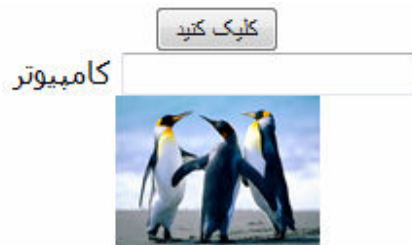
اگر برنامه را اجرا کنید عکس را مشاهده خواهید کرد.

نکته : اگر خواستید عکس یا هر کنترل دیگر قسمتی از صفحه را اشکال کند و با بزرگ و کوچک شدن نسبت به اندازه صفحه تغییر کند از علامت % استفاده می کنیم مثلا اگر Width را برابر ۵۰ درصد قرار بدهیم یعنی ۵۰ درصد صفحه را به نوعی اشغال میکند .

```

<asp:Image Width="50%" ID="Image1" runat="server"
ImageUrl="~/NewFolder1/Penguins.jpg" />

```



قبل از ادامه بحث ها با چند مورد از کنترل های html آشنا می شویم برای اینکار یک برنامه جدید وب ایجاد کنید.

`<hr />` : برای ایجاد خط از این کنترل استفاده می شود. مثال:

```
<div >
  <hr />
</div>
```

`<br />` : برای انتقال به سطر پایین.

`<h1>` : برای نمایش متن با ضخامت متفاوت که از یک شروع شده و تا ۶ ادامه می یابد. مثال:

```
<div >
  <hr />
  <h1>Computer</h1>
  <h2>Computer</h2>
  <h3>Computer</h3>
  <h4>Computer</h4>
  <h5>Computer</h5>
  <h6>Computer</h6>
</div>
```

**Computer**

**Computer**

**Computer**

**Computer**

**Computer**

**Computer**

`<B>` : برای پررنگ کردن متن.

`<B>`This text displays boldface.`</B>`

`<I>`This text is italic.`</I>`

`<ol>` : برای شماره گذاری متن. مثال:

```
<ol>
<li>computer</li>
<li>book</li>
<li>door</li>
```





```
</div>
```

نکته : ستون های جدول داخل تگ سطر قرار می گیرد برای مثال در کد زیر سه ستون به جدول اضافه می کنیم

:

```
<table border="2">
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</table>
```



یک سطر دیگر به برنامه اضافه می کنیم که دارای سه ستون است :

```
<table border="2">
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</table>
```

برای اینکه جدول ما کل عرض صفحه را در بر گیرد `width` آن را برابر `100%` قرار می دهیم:

```
<table border="2" width="100%">
```


حال دو تکست باکس و دو لیبل بر روی ستون های اول و دوم قرار دهید و تکست لیبل ها را به نام و پسورد

تغییر دهید و یک دکمه بر روی ستون سوم با تکس ورود اضافه کنید:

UserNmae	<input type="text"/>	
Password	<input type="text"/>	ورود

حال برنامه را اجرا کنید و نتیجه را مشاهده کنید. همانطور که خواهید دید کنترل ها با نظم خاصی بر روی صفحه چیده شده اند. برای اینکه هر ستون دارای اندازه خاص خود باشد بهتر است width آنها را به شکل زیر تغییر دهید:

```
<table border="2" width="100%">
  <tr>
    <td style="width:20%" >
      <asp:Label ID="Label1" runat="server" Text="UserNmae"></asp:Label>
    </td>
    <td style="width:50%">
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
    </td>
    <td style="width:30%">برای کلیک ورود برای</td>
  </tr>
  <tr>
    <td style="width:30%" >
      <asp:Label ID="Label2" runat="server" Text="Password"></asp:Label>
    </td>
    <td style="width:50%">
      <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
    </td>
    <td style="width:30%">
      <asp:Button ID="Button1" runat="server" Text="ورود" />
    </td>
  </tr>
</table>
```

UserNmae	<input type="text"/>	برای ورود کلیک کنید
Password	<input type="text"/>	ورود

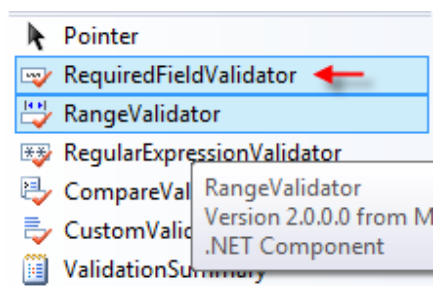
اگر خواستید لبه های جدول نمایش داده نشود می توانید "border="2" را حذف کنید:

UserNmae	<input type="text"/>	برای ورود کلیک کنید
Password	<input type="text"/>	ورود

برنامه را اجرا کنید و بر روی دکمه ورود کلیک کنید مشاهده خواهید کرد که هیچ عملی را انجام نمی دهد چون هیچ کدی برای آن نوشته نشده است .

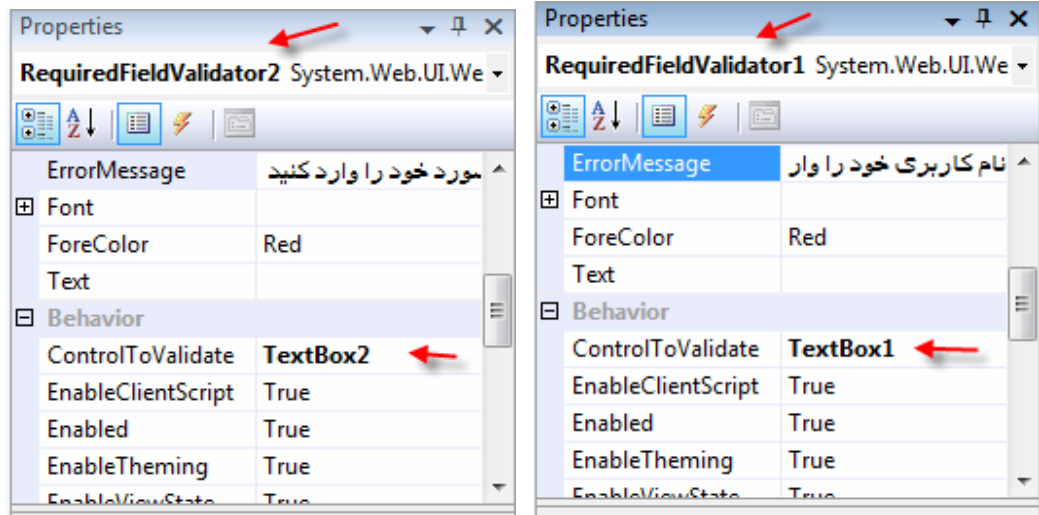
## : Validation

اگر در مثال بالا خواستید کاربر حتما نام کاربری و پسورد خود را وارد کند در غیر این صورت وقتی دکمه ورود را کاربر می زند هیچ عملی انجام نگیرد در این صورت می توانید به دو روش عمل کنید روش اول استفاده از شرط هاست که این اصلا کار خوبی نیست چون وقتی شرطی می نویسید این شرط به سرور فرستاده می شود و سرور این شرط را کنترل می کند و به ما پاسخ می دهد که هم زمان بیشتری می خواهد و هم وقت سرور را بیخودی می گیریم بنابراین قبل از اینکه بخواهیم چیزی به سرور فرستاده شود این کار را کنترل کنیم باید از Validation در سمت Client (کاربر-مقاضی) استفاده کنیم . برای درک بیشتر این روش در مثال بالا دو کنترل `RequiredFieldValidator` در کنار دو تکست باکس قرار دهید و `ErrorMessage` این دو کنترل را با رشته دلخواه خود پر کنید مثلا لطفا ابتدا رمز خود را وارد کنید.



UserNmae	<input type="text"/>	نام کاربری خود را وارد کنید
Password	<input type="password"/>	ابتدا پسورد خود را وارد کنید
	<input type="button" value="ورود"/>	

حال باید این ها را انتصاب بدهیم به تکست باکس ها اولی را به `TextBox1` یک و دومی را به `TextBox2` دو منتصب می کنیم مانند شکل زیر



```

<table border= width="100%" >
  <tr>
    <td style="width:20%" >
      <asp:Label ID="Label1" runat="server" Text="UserNmae"></asp:Label>
    </td>
    <td style="width:50%">
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
      <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server"
      ControlToValidate="TextBox1" ErrorMessage="نام
نام کاربری خود را وار"></asp:RequiredFieldValidator>
    </td>
    <td style="width:30%">کنید کلیک ورود برای</td>
  </tr>
  <tr>
    <td style="width:30%" >
      <asp:Label ID="Label2" runat="server" Text="Password"></asp:Label>
    </td>
    <td style="width:50%">
      <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
      <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
runat="server"
      ControlToValidate="TextBox2" ErrorMessage=" ابتدا
ابتدا خود پسورد"></asp:RequiredFieldValidator>
    </td>
    <td style="width:30%">
      <asp:Button ID="Button1" runat="server" Text="ورود" />
    </td>
  </tr>
</table>

```

حال برنامه را اجرا کنید و بر روی دکمه ورود کلیک کنید و یک بار هم رشته ای را در تکست باکس ها وارد کنید و نتیجه را مشاهده کنید.

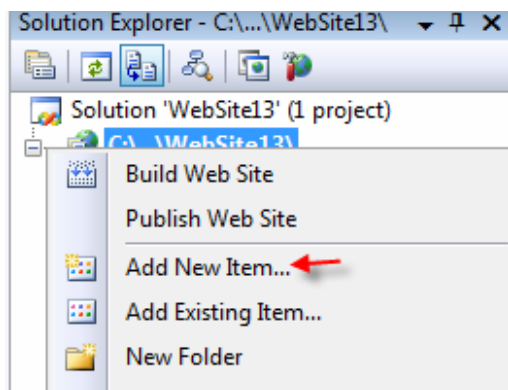
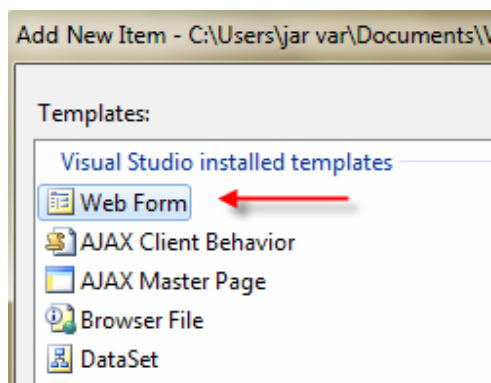
UserNmae	ms	برای ورود کلیک کنید
Password		ابتدا پسورد خود را وارد کنید <input type="button" value="ورود"/>

اگر خواستید علاوه بر خالی بودن شرط دیگری را نیز چک کنید مثلا عدد وارده بزرگتر از ۱۳۷۷ نباشد از کنترل RangeValidator استفاده می شود. مثال:

```
<asp:RangeValidator ID="RangeValidator1" runat="server"
    ControlToValidate="TextBox1" ErrorMessage="خطا"
    MaximumValue="1377"
    MinimumValue="1357">خطا</asp:RangeValidator>
```

اضافه کردن صفحات :

برای اضافه کردن صفحات دیگر به برنامه کفایت از قسمت Add new Item یک Web Form با نام دلخواه به برنامتون اضافه کنید



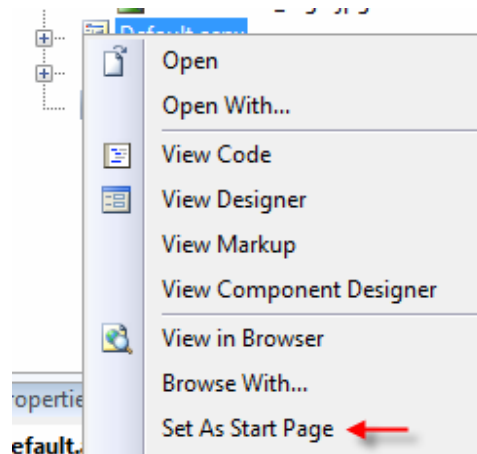
بر روی صفحه جدید یک تکست باکس قرار دهید

```
<body>
    <form id="form1" runat="server">
        <div>
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        </div>
    </form>
</body>
</html>
```

و در صفحه اول بر روی دکمه ورود دوبار کلیک کنید و کد زیر را برای آن بنویسید .

```
protected void Button1_Click(object sender, EventArgs e)
{
    Response.Redirect ("Default2.aspx");
}
```

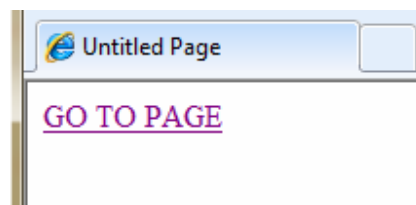
اما قبل از اینکه برنامه را اجرا کنید باید صفحه اول را به عنوان اولین صفحه ای قرار دهید که قرار است در سرور اجرا شود برای اینکار کافیست بر روی صفحه اول کلیک راست کنید و `set as start page` آن را انتخاب کنید.



حال برنامه را اجرا کنید و مقادیر تکست باکس را پر کنید و بر روی دکمه ورود کلیک کنید. کار این کد هدایت به صفحات دیگر است .

برای هدایت به صفحات دیگر از کدهای html نیز می توانید استفاده کنید برای مثال کد زیر به صفحه مشخص شده هدایت می شود:

```
<div>
  <a href="Default2.aspx">GO TO PAGE</a>
</div>
```



نکته : وقتی شما یک متغیر محلی تعریف می کنید و در کنترلی آن را مقدار دهی می کنید و به سرور می فرستید دیگر وقتی از سرور برگشت مقدارش از بین می رود .

نکته : برای انتقال اطلاعات بین صفحات از Session استفاده می شود چون Session در سمت سرور ذخیره می شود و اطلاعات موقع برگشت پاک نمی شود. مثلا در مثال بالا می خواهیم مقدار تکست باکسهای صفحه اول را در صفحه دوم نمایش دهیم برای اینکار در رویداد دکمه ورود کد زیر را بنویسید:

```
protected void Button1_Click(object sender, EventArgs e)
{
    Session.Add("mas", TextBox2.Text + TextBox1.Text);
    Response.Redirect("Default2.aspx");
}
```

وقتی بخواهید مقداری را به وسیله Session بفرستید باید یک نام برای آن ذکر کنید تا با آن نام به مقادیر Session دستیابید. حال در صفحه دوم دکمه ای قرار دهید

```
<body>
  <form id="form1" runat="server">
    <div>
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
      <br />
      <asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="ts" />
    </div>
  </form>
</body>
```

و کد زیر را برای آن بنویسید تا مقدار Session را مشاهده کنید:

```
protected void Button1_Click(object sender, EventArgs e)
{
    TextBox1.Text = Session["mas"].ToString();
}
```

ب

اگر خواستید به هنگام فراخوانی صفحه دوم نیز می توانید مقادیری بفرستید مثل کد زیر برای دکمه صفحه اول:

```
protected void Button1_Click(object sender, EventArgs e)
```



```

{
    Session.Add("mas", TextBox2.Text + TextBox1.Text);
    Response.Redirect("Default2.aspx?num1=" + TextBox1.Text + "&num2=" +
    TextBox2.Text);
}

```

در این صورت مقدار TextBox1 در num و دومی در num2 قرار میگیرد. حال کد صفحه دوم را بنویسید: برای امتحان دو لیبل بر روی صفحه دوم قرار دهید.

```

protected void Button1_Click(object sender, EventArgs e)
{
    TextBox1.Text = Session["mas"].ToString();
    Label1.Text = Request.QueryString["num1"];
    Label2.Text = Request["num2"];
}

```

بر نامه را اجرا کنید و نتیجه را مشاهده کنید.

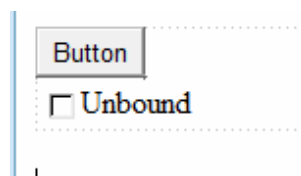
نکته : می توانید QueryString را حذف کنید.

یک پروژه جدید ایجاد کنید و یک دکمه و یک CheckBoxList بر روی صفحه قرار دهید

```

<body>
<form id="form1" runat="server">
<div>
<asp:Button ID="Button1" runat="server" Text="Button" />
<br />
<asp:CheckBoxList ID="CheckBoxList1" runat="server">
</asp:CheckBoxList>
</div>
</form>
</body>

```



هدف از این برنامه اضافه کردن CheckBox به تعداد دلخواه به CheckBoxList1 است و یک تعداد کارهای دیگر که در ادامه به آنها خواهیم رسید. برای انجام اینکار رویداد کلیک دکمه را فراخوانی کنید و کد زیر را در آن بنویسید:

```

protected void Button2_Click(object sender, EventArgs e)
{

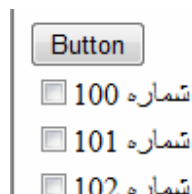
```

```

ListItem g;
for (int i = 100; i <= 150; i++)
{
    g = new ListItem();
    g.Text = "شماره " + i;
    g.Value = i.ToString();
    CheckBoxList1.Items.Add(g);
}
}

```

برای اضافه کردن به `CheckBoxList1` کفایت یک شی از کلاس `ListItem` ایجاد کنید (به تعداد `CheckBox`) و `Text` و `Value` آن را مقدار دهی کنیم و به آیتمهای `CheckBoxList1` اضافه کنیم. (چون با استفاده از `value` به عناصر `CheckBoxList1` می توانیم دسترسی داشته باشیم) حال برنامه را اجرا کنید و نتیجه را مشاهده کنید:



اگر خواستید `CheckBox` ها در چند ستون نمایش داده شود می توانید `RepeatColumns` به تعداد دلخواه تنظیم کنید مثلا اگر `RepeatColumns` را برابر سه قرار دهید کل `CheckBox` ها در سه ستون نمایش داده می شود.

```

<asp:CheckBoxList ID="CheckBoxList1" runat="server" RepeatColumns="3">
</asp:CheckBoxList>

```

یک `RadioButtonList` و یک دکمه دیگر به برنامه اضافه کنید می خواهیم کاربر چک باکس هایی را که انتخاب کرده (تیک زده) در یک `RadioButtonList` نمایش داده شود. برای دکمه دوم کد زیر را بنویسید:

```

protected void Button2_Click(object sender, EventArgs e)
{
    for (int i = 0; i <= 50; i++)
    {
        ListItem g;

        if (CheckBoxList1.Items[i].Selected)
        {
            g = new ListItem();
            g.Text = "شماره " + CheckBoxList1.Items[i].Text;
            g.Value = (i).ToString();

```

```

        RadioButtonList1.Items.Add(g);
    }
}

```

برنامه را اجرا کنید و چند چک باکس انتخاب کنید و بر روی دکمه دوم کلیک کنید:



اگر بخواهید با کلیک بر روی دکمه اول پیغامی به شما نمایش داده شود مبنی بر اینکه آیا مایلید این کار انجام گیرد باید از جاوا اسکریپت ها استفاده کنید برای انجام اینکار کد زیر را در قسمت SOURCE بنویسید:

```

<head runat="server">
    <title>Untitled Page</title>
    <script type="text/jscript" language="javascript">
        function aa()
        {
            if(!confirm('شما اطلاعات مایلید آیا'))
            {
                return false;
            }
        }
    </script>
</head>

```

و در دکمه اول `OnClick` برابر با نام تابع جاوا اسکریپت قرار دهید و برنامه را اجرا کنید

```

<asp:Button ID="Button1" runat="server" Text="Button" onclick="Button1_Click"
    OnClientClick=" return aa()" />

```

<br />



روش کار به این صورت است که قبلا از ایکه متد `onclick` اجرا شود `OnClickClicked` اجرا می شود(در سمت client) و در `OnClickClicked` جاوا اسکریپت فراخوانی می شود اگر کاربر بر روی دکمه `ok` کلیک کرد تابع `aa` یک مقدار `true` برگشت می دهد وگرنه مقدار `false` برگشت داده می شود و متد `onclick` اجرا نمی شود (یعنی اطلاعات به سمت سرور فرستاده نمی شود).

اگر خواستید با کلیک بر روی هر `RadioButton` تکست آنها در پیغامی نمایش داده شود باید بازهم از جاوا اسکریپت استفاده کنید ولی چگونه تکست `RadioButton` را به جاوا اسکریپت انتقال دهیم؟

برای اینکار یک متغیر عمومی تعریف می کنیم و رویداد

`RadioButtonList1_SelectedIndexChanged`

را با دو بار کلیک بر روی `RadioButtonList1` فراخوانی می کنیم:

```
public string str;
protected void RadioButtonList1_SelectedIndexChanged(object sender,
EventArgs e)
{
}
```

و کد جاوا اسکریپت را در قالب استرینگ در این رویداد می نویسیم و آن را در `str` قرار داده و در قسمت `source` از آن استفاده می کنیم:

```
public string str;
protected void RadioButtonList1_SelectedIndexChanged(object sender,
EventArgs e)
{
    str = "<script language=javascript>alert('" +
    RadioButtonList1.SelectedItem.Text + "')</script>";
}
```

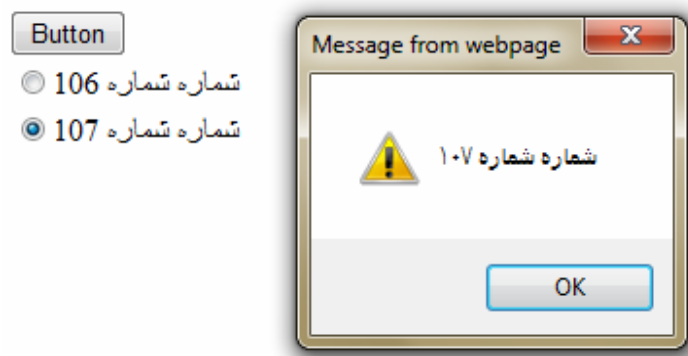
برای قرار دادن این استرینگ در قسمت source کافیست به این شکل عمل کنیم

```
</div>  
<%=str %>  
</form>
```

قبل از اجرای برنامه `AutoPostBack` و `RadioButtonList1` و `CheckBoxList` را به `true` تغییر دهید  
این کار باعث می شود تا با هر بار کلیک بر روی آنها کدهای مربوط به آنها در سمت سرور اجرا شود:

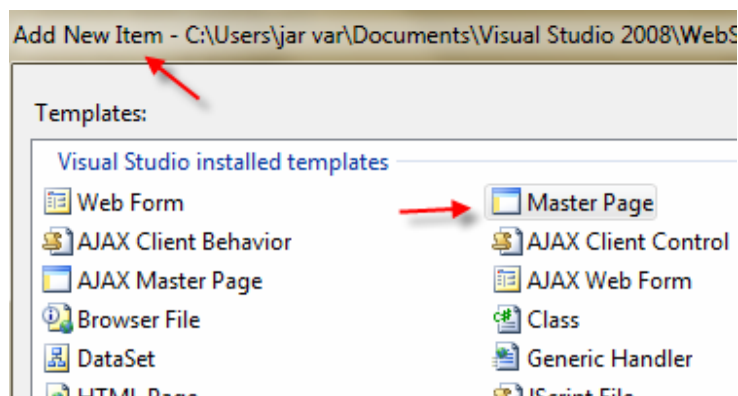
```
<asp:RadioButtonList ID="RadioButtonList1" runat="server"  
    onselectedindexchanged="RadioButtonList1_SelectedIndexChanged"  
    AutoPostBack="True">  
  
</asp:RadioButtonList>
```

برنامه را اجرا کنید و نتیجه را مشاهده کنید:



## :MasterPage

اگر خواستید همه صفحات دارای یک قسمت ثابت باشد باید از `MasterPage` استفاده کنید مثلا اگر خواستید همه صفحات دارای یک عکس در بالای صفحه و یک منو در سمت راست باشد باید از `MasterPage` ها استفاده کنید اگر از `MasterPage` ها استفاده نکنید باید در هر صفحه دوباره از اول مراحل طراحی قسمت های ثابت را برای هر فرم انجام بدهید. برای آشنایی بیشتر یک برنامه جدید وب ایجاد کنید و از `Add New Item` یک `MasterPage` به برنامه اضافه کنید



حال اگر در صفحه MasterPage هر گونه طراحی کنید و صفحات را به آن نسبت بدهید همه صفحات در قسمتی از صفحه اشتراکی را دارند که در MasterPage طراحی شده است. در MasterPage قسمتی وجود با نام `ContentPlaceholder` که برای طراحی جداگانه فرمها استفاده می شود یعنی قسمتی که در صفحات مختلف مشترک نیستند و کنترل های هر صفحه در آن قرار می گیرد.

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="MasterPage.master.cs" Inherits="MasterPage" %>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Untitled Page</title>
  <asp:ContentPlaceholder id="head" runat="server">
  </asp:ContentPlaceholder>
</head>
<body>
  <form id="form1" runat="server">
  <div>
    <asp:ContentPlaceholder id="ContentPlaceholder1"
runat="server">

    </asp:ContentPlaceholder>
  </div>
  </form>
</body>
</html>
```

بنابراین اگر جدولی در MasterPage ایجاد کنید و در یک قسمت جدول `ContentPlaceholder` قرار دهیم آن قسمت جدول به انواع فرمها برای قرار دادن کنترل ها اختصاص داده می شود.

برای مثال در همین قسمت برنامه یک جدول ۲×۲ ایجاد می کنیم

```

<form id="form1" runat="server">
<div>
  <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

  </asp:ContentPlaceHolder>

  <table border="1" width="100%">
    <tr>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <td></td>
      <td></td>
    </tr>
  </table>

</div>
</form>

```


نکته : برای اینکه دو ستون بالایی جدول را به هم متصل کنیم باید یکی از tdها را در tr حذف کنیم و از `colspan="2"` استفاده کنید:

```

<table border="1" width="100%">
  <tr>
    <td colspan="2"></td>
  </tr>
  <tr>
    <td ></td>
    <td></td>
  </tr>
</table>

```


بعد `ContentPlaceHolder` را داخل سطر دوم و ستون اول قرار دهید تا هر صفحه کنترل های مورد نیاز خود را در این قسمت قرار دهد.

```

<table border="1" width="100%">
  <tr>
    <td colspan="2"></td>
  </tr>
  <tr>

```

```

        <td ><asp:ContentPlaceHolder id="ContentPlaceHolder1"
runat="server">
        </asp:ContentPlaceHolder>
        </td>
        <td></td>
    </tr>
</table>

```



حال به سطر اول یک عکس قرار می دهیم تا در هر صفحه قرار گیرد روش قرار دادن عکس را در قسمت های قبلی توضیح دادیم.

```

<table border="1" width="100%">
<tr>
<td colspan="2">
<asp:Image ID="Image1" runat="server"
ImageUrl="~/NewFolder1/Persia 176.jpg" />
</td>
</tr>
<tr>
<td ><asp:ContentPlaceHolder id="ContentPlaceHolder1"
runat="server">
</asp:ContentPlaceHolder>
</td>
<td></td>
</tr>
</table>


```



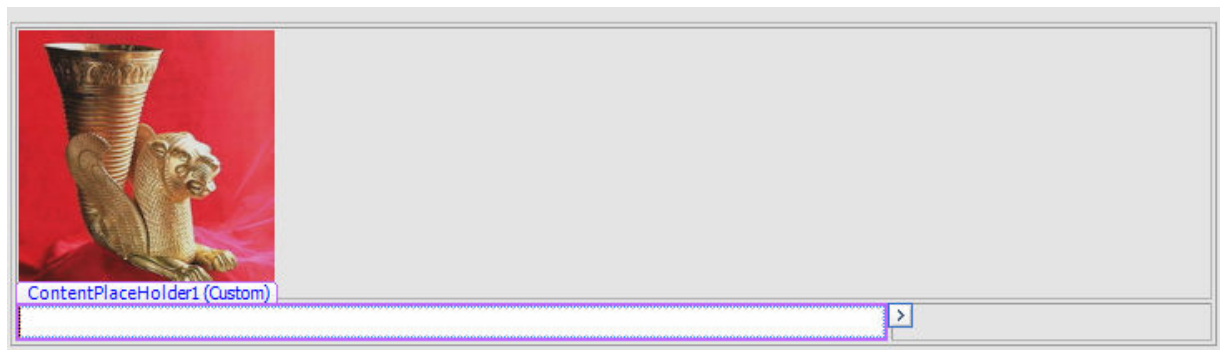
بعد از اینکه طراحی صفحه master page را انجام دادیم دیگر به این صفحه نیاز نداریم .

برای متصل کردن صفحاتمان به master page کافیست موقع اضافه کردن یک صفحه جدید گزینه select master page را تیک بزنیم .



- Place code in separate file
- Select master page 

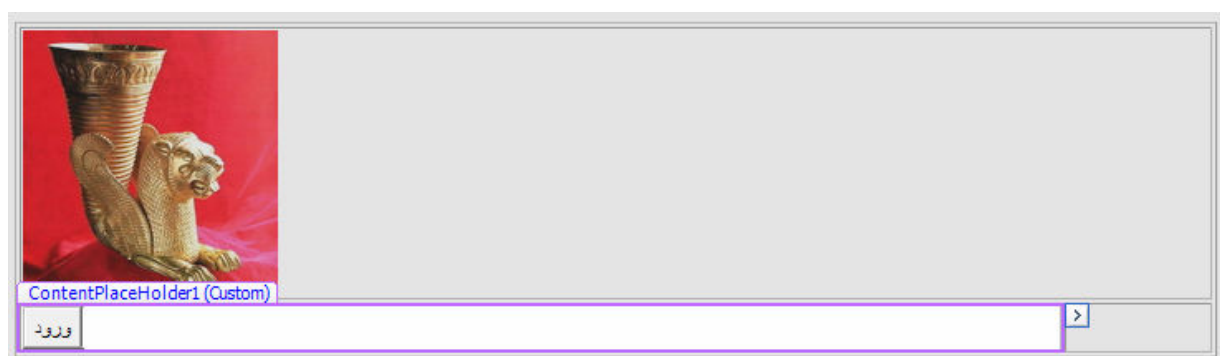
برای امتحان صفحه قبلی را حذف کنید و دو صفحه جدید به برنامه اضافه کنید



و در قسمت `ContentPlaceHolder` هر صفحه کنترل های مورد نیاز را قرار دهید (سعی کنید کنترل هر صفحه متفاوت باشد برای فهمیدن تفاوت صفحات)

در صفحه اول یک دکمه و در صفحه دوم یک تکست باکس قرار دهید و کد زیر را برای دکمه صفحه اول قرار بنویسید:

```
protected void Button1_Click(object sender, EventArgs e)
{
    Response.Redirect("Default2.aspx");
}
```



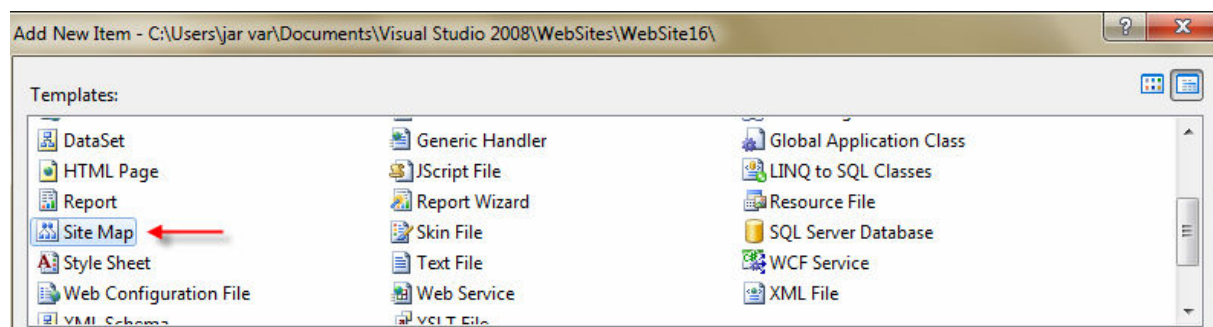
و صفحه اول را به عنوان `Set As start page` انتخاب کنید و برنامه را اجرا کنید و نتیجه را مشاهده کنید.

برای قرار دادن عکس در وسط صفحه باید `text-align` آن قسمت از جدول را به `center` تغییر دهید:

```
<tr>
    <td colspan="2" style="text-align:center">
        <asp:Image ID="Image1" runat="server"
        ImageUrl="~/NewFolder1/Persia 176.jpg" />
    </td>
```



حال به طراحی منوی سطر دوم و وستون دوم می پردازیم هدف از این منوها هدایت به بخش های مختلف (صفحات)سایتمان است برای اینکار از Navigation ها استفاده می کنیم در Navigation چندین کنترل وجود دارد که از آنها استفاده می کنیم مثل Menu و TreeView. روش کار به این صورت است که در ابتدا یک sitemap به برنامه اضافه می کنیم

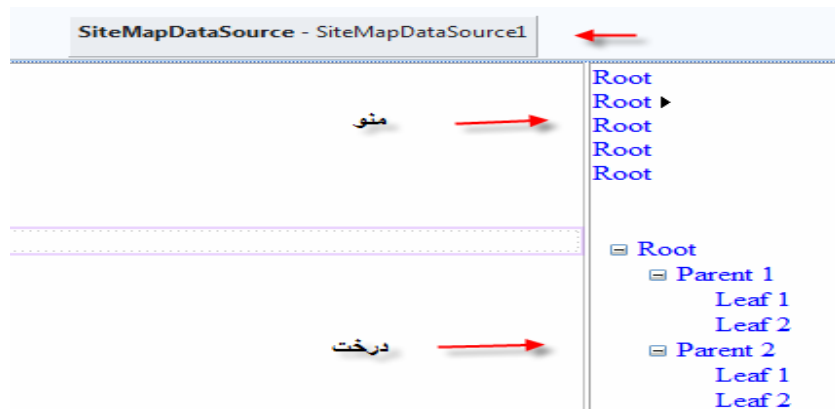


```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="" title="" description="">
        <siteMapNode url="" title="" description="" />
        <siteMapNode url="" title="" description="" />
    </siteMapNode>
</siteMap>
```

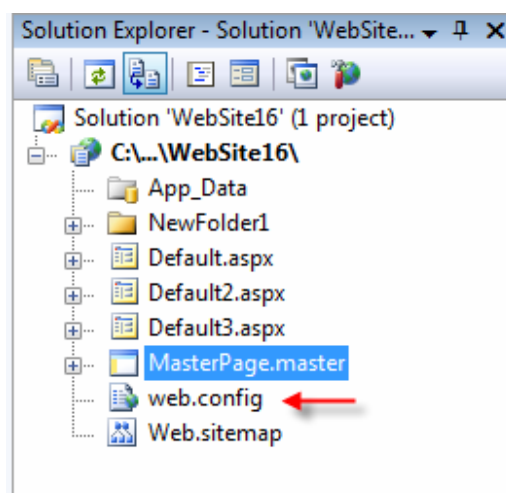
بعد نام صفحه اصلی و صفحات دیگر را که به عنوان شاخه استفاده می شود را در `url` می نویسیم و در `title` توضیحاتی در باره صفحات می دهیم مثلا صفحه ثبت نام و یا صفحه کارنامه.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="Default.aspx" title="اصلي صفحه" description="">
    <siteMapNode url="Default2.aspx" title="تب نام" description="" />
    <siteMapNode url="Default3.aspx" title="کارنامه" description="" />
  </siteMapNode>
</siteMap>
```

بعد یک منو و یک درخت در خانه مشخص شده جدول قرار میدهیم و یک `SiteMapDataSource` نیز در صفحه برناممون قرار می دهیم:



بعد صفحه `web.config` را باز کرده :



و کد زیر را بین `</system.web>` `<system.web>` بنویسید:

```
<siteMap>
  <providers>
    <add name="MYU" type="System.Web.XmlSiteMapProvider"
siteMapFile="Web.sitemap"></add>
  </providers>
</siteMap>
</system.web>
```

`SiteMapDataSource` در قسمت `SiteMapProvider` به این کد بالا نیاز دارد و کد بالا نیز به یک `Web.sitemap` نیاز داشت که اضافه کردیم بنابراین کد زیر را برای `SiteMapDataSource` اضافه می کنیم:

```
<asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server"
  SiteMapProvider="MYU" />
```

بعد `DataSourceID` کنترل های منو و درخت را برابر `SiteMapDataSource1` قرار می دهیم و برنامه را اجرا می کنیم:

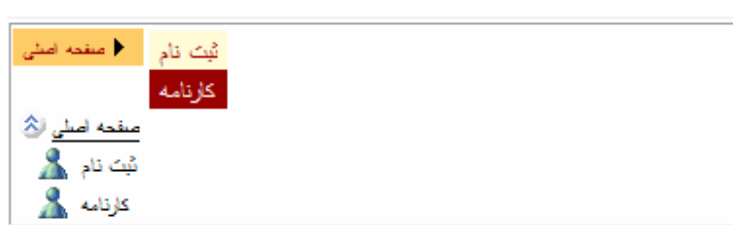
```
<asp:Menu ID="Menu1" runat="server"
DataSourceID="SiteMapDataSource1">
</asp:Menu>
<br />
<asp:TreeView ID="TreeView1" runat="server"
DataSourceID="SiteMapDataSource1" >
</asp:TreeView>
```



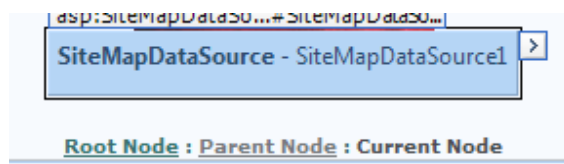
برای زیبایی منو و درخت می توانید بر روی علامت `<` کلیک کنید و `Auto Format` آن را به شکل دلخواه تنظیم کنید:



برنامه را اجرا کنید نتیجه را مشاهده کنید:



از کنترل SiteMapPath نیز می توانید به راحتی و با قرار دادن آن بر روی صفحه از آن استفاده کنید:



از این به بعد هر صفحه ای به برنامه اضافه کردید کافایت آن را در Web.sitemap معرفی کنید برای مثال یک صفحه با نام computer به برنامه اضافه می کنیم و کد را به شکل زیر تغییر می دهیم:

```
?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
```

```

<siteMapNode url="Default.aspx" title="اصلي صفحه " description="">
  <siteMapNode url="Default2.aspx" title="نام ثبت " description="" />
  <siteMapNode url="Default3.aspx" title="کارنامه" description="" />
  <siteMapNode url="computer.aspx" title="کامپیوتر"
description="" />
</siteMapNode>
</siteMap>

```

اگر خواستید صفحات را در شاخه دیگر قرار دهید بهتر است یک `<siteMapNode></siteMapNode>` دیگر تعریف کنید و صفحات مورد نظر را داخل آن قرار دهید ولی باید به این نکته توجه کنید که این شاخه باید داخل شاخه قبلی قرار گیرد.

```

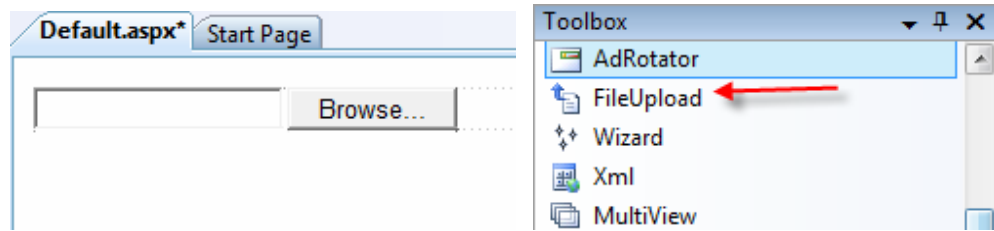
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="Default.aspx" title="اصلي صفحه " description="">
    <siteMapNode url="Default2.aspx" title="نام ثبت" description="" />
    <siteMapNode url="Default3.aspx" title="کارنامه" description="" />
    <siteMapNode url="computer.aspx" title="کامپیوتر" description="" />
    <siteMapNode url="a.aspx" title="فیلم" description="">
      <siteMapNode url="b.aspx" title="سریال " description="" />
    </siteMapNode>
  </siteMapNode>
</siteMap>

```



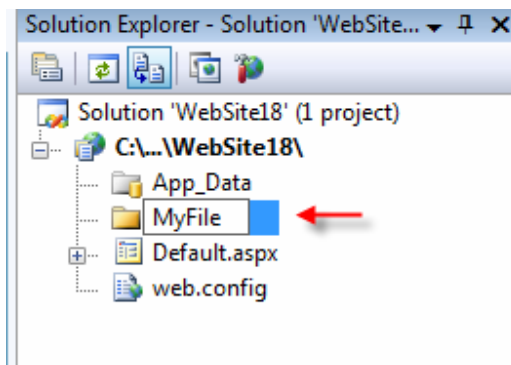
## : FileUpload

اگر خواستید از طرف کاربر فایل را دریافت کنید و در سرور ذخیره کنید می توانید از FileUpload استفاده کنید. برای آشنایی با این کنترل یک برنامه جدید وب ایجاد کنید واز کنترل FileUpload بر روی صفحه قرار دهید:



```
<div>
  <asp:FileUpload ID="FileUpload1" runat="server" />
</div>
```

یک دکمه و یک لیست باکس نیز بر روی صفحه قرار دهید و یک پوشه نیز با نام دلخواه در قسمت solution Explorer ایجاد کنید. (برای ذخیره فایلها).



```
<div>
  <asp:FileUpload ID="FileUpload1" runat="server" />
  <br />
  <asp:Button ID="Button1" runat="server" Text="Save" />
  <br />
  <asp:ListBox ID="ListBox1" runat="server"></asp:ListBox>
</div>
```

حال کد زیر را برای دکمه ذخیره بنویسید می خواهیم اول بعضی مشخصات فایل انتخاب شده توسط کاربر را در لیست باکس نمایش دهیم (مثل اندازه فایل و فرمت) و بعد فایل را (مثل عکس) در پوشه ذخیره کنیم.

```
protected void Button1_Click(object sender, EventArgs e)
{
    ListBox1.Items.Add(FileUpload1.PostedFile.ContentLength.ToString());
    ListBox1.Items.Add(FileUpload1.PostedFile.ContentType);
    ListBox1.Items.Add(FileUpload1.PostedFile.FileName);
    ListBox1.Items.Add(FileUpload1.HasFile.ToString());
    ListBox1.Items.Add(Request.PhysicalApplicationPath.ToString());
    string str = Request.PhysicalApplicationPath + "MyFile\\" +
FileUpload1.FileName;
    ListBox1.Items.Add(str);
    FileUpload1.SaveAs(str);
}
```

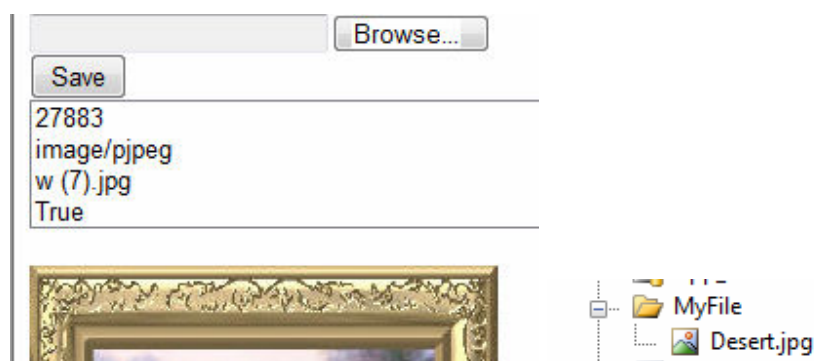
}

هدف از به دست آوردن مشخصات فایل اینست که اگر خواستید می توانید فایلهای مورد نظر را از کاربر بخواهید و محدود کنید مثلا اندازه فایلها کوچکتر از ۱ کیلوبایت نباشد و غیره . اگر فایل شما عکس بود و خواستید نمایش بدهید بهتر است یک کنترل `img` از نوع `html` و یا `image` از نوع `asp` بر روی صفحه قرار دهید و کد بالا را به زیر تغییر دهید.(اضافه کنید):

```
<br />  
<img alt="" id="mas" runat="server" src="" />
```

```
mas.Src = "~/MyFile/" + FileUpload1.FileName;
```

برنامه را اجرا کنید و از قسمت **Browse** یک عکس انتخاب کنید و دکمه **Save** را بزنید و نتیجه را مشاهده کنید بعد می توانید عکس مورد نظر را که در پوشه ذخیره شده را نیز ببینید.



نکته : اگر `FileUpload1.HasFile` مقدار `true` را برگرداند یعنی کاربر یک فایل را انتخاب کرده در غیر این صورت مقدار `false` را بر می گرداند.

## AJAX

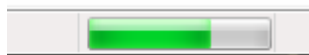
یکی از جدید ترین تکنولوژی ها در زمینه وب سایت تکنولوژی `ajax` است روش کار به این صورت است که وقتی شما درخواستی را به سرور می فرستید اگر از `ajax` استفاده کنید دیگر کل محتویات بسته شما به سرور فرستاده نمی شود بلکه آن قسمتی به سرور فرستاده می شود که نیاز دارید و این باعث افزایش سرعت



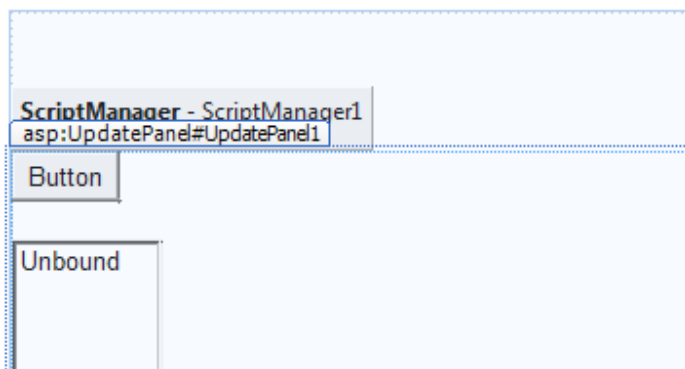
وب سایت شما خواهد شد . برای درک این مساله یک برنامه تحت وب ایجاد کنید و یک دکمه و یک لیست باکس بر روی صفحه قرار دهید و کد زیر را برای دکمه بنویسید و برنامه را اجرا کنید:

```
protected void Button1_Click(object sender, EventArgs e)
{
    for (int i = 0; i < 1000; i++)
    {
        ListBox1.Items.Add("computer" + i);
    }
}
```

بر روی دکمه کلیک کنید تا لیست باکس شروع به پر شدن کند در این لحظه اگر به پایین صفحه نگاه کنید متوجه پر شدن یک **Progress Bar** بار تا پایان انجام کار خواهید بود



این کار ممکن است در صفحات مختلف دارای زمانهای متفاوتی باشد. حال می خواهیم این کار را با **ajax** انجام بدیم تا شاهد تفاوت آنها شوید برای اینکار یک کنترل **ScriptManager** و یک **UpdatePanel** در صفحه قرار دهید و کنترل های قبلی را در داخل **UpdatePanel** قرار دهید(مطابق شکل)



```
<asp:ScriptManager ID="ScriptManager1" runat="server">
</asp:ScriptManager>

<asp:UpdatePanel ID="UpdatePanel1" runat="server">
    <ContentTemplate>
        <asp:Button ID="Button1" runat="server"
onclick="Button1_Click"
Text="Button" />
        <br />
        <br />
        <asp:ListBox ID="ListBox1" runat="server"></asp:ListBox>
    </ContentTemplate>
</asp:UpdatePanel>
```

```
</ContentTemplate>  
</asp:UpdatePanel>
```

```
</div>
```

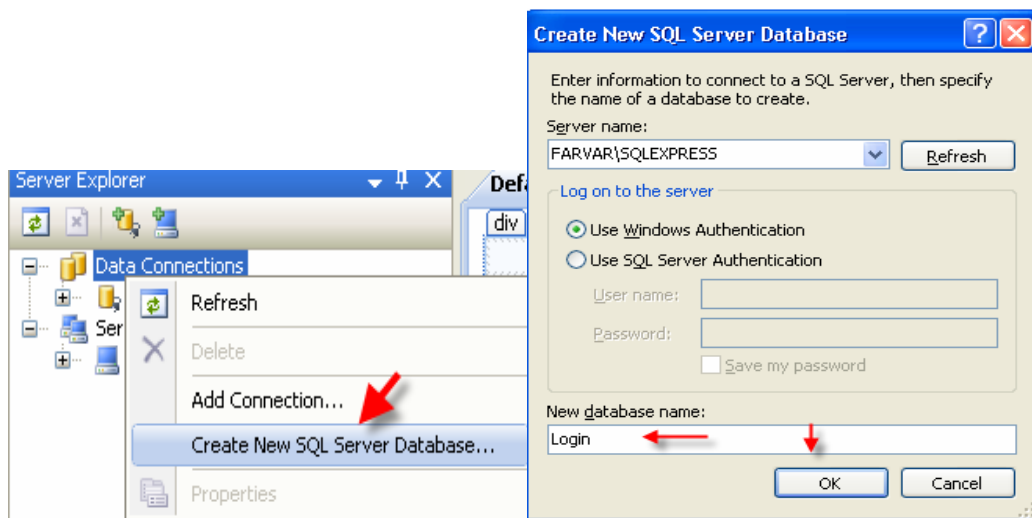
حال برنامه را اجرا کنید و روی دکمه کلیک کنید تا نتیجه را مشاهده کنید. با این کار فقط دستورات مربوط به کنترل لیست باکس به سرور فرستاده می شود و دیگر قسمت های صفحه فرستاده نمی شود و این باعث افزایش سرعت می شود.

## کار با پایگاه داده SQL

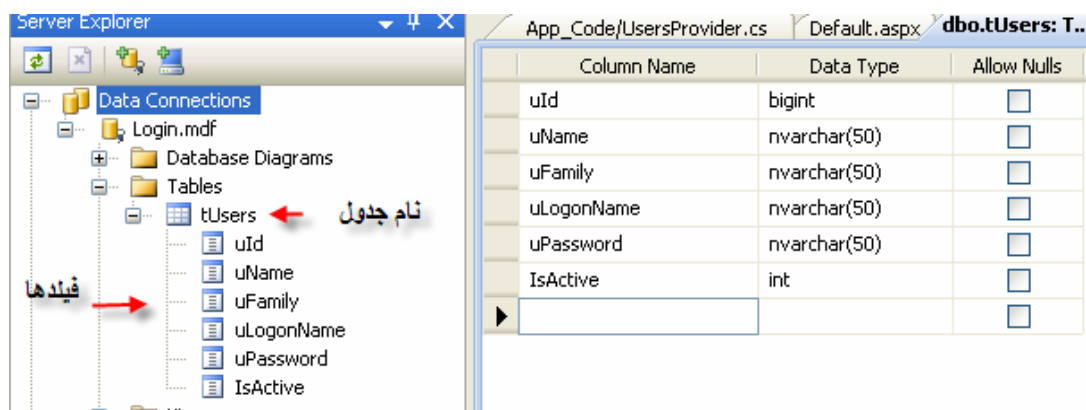
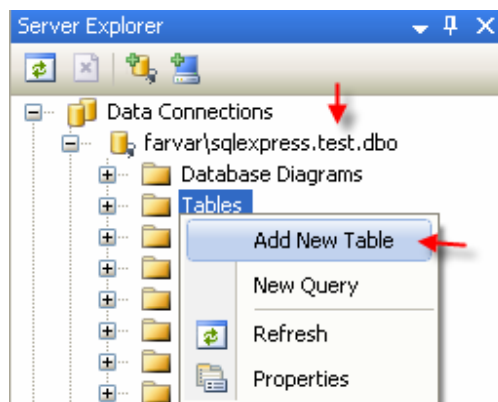
در بخش های قبلی با پایگاه داده های مختلف و نحوه ارتباط با آنها بحث کردیم در این قسمت می خواهیم تا روش دیگری نیز برای ارتباط با پایگاه داده بیان کنیم و آن استفاده از `ObjectDataSource` است این کنترل فقط با استفاده از `PROCEDURE` ها با پایگاه داده `sql server` ارتباط برقرار می کند که این کار باعث افزایش امنیت برنامه می شود چون وقتی از `PROCEDURE` استفاده می کنیم دستورات زبان `sql` را به داخل `sql server` منتقل می کنیم و در داخل `PROCEDURE` با نام دلخواه ذخیره می کنیم و فقط نام آن را داخل سی شارپ فراخوانی می کنیم بنابراین اگر کسی سایت ما را هک کرد دیگر هیچ چیز درباره کدمان نمی داند.

مثال :

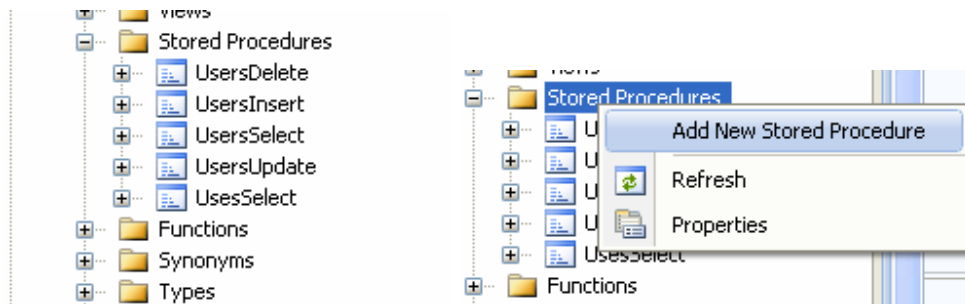
یک برنامه وب جدید ایجاد کنید ایجاد کنید و از قسمت `Server Explorer` یک پایگاه داده جدید با نام `test` ایجاد کنید .



بعد یک جدول با نام دلخواه ایجاد کنید و فیلدهای آن را نیز مطابق شکل زیر یا دلخواه ایجاد کنید



بعد پروسیجرهای لازم را ایجاد کنید



```
ALTER PROCEDURE dbo.UsersDelete
```

```
    @UId bigint
```

```
AS
```

```
    DELETE FROM tUsers
```

```
    WHERE      (uId = @UId)
```

```
    RETURN
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
ALTER PROCEDURE dbo.UsersInsert
```

```
    @uName  nvarchar(50),
```

```
    @uFamily nvarchar(50),
```

```
    @uLogonName  nvarchar(50),
```

```
    @uPassword  nvarchar(50),
```

```
    @IsActive  int
```

```
AS
```

```
    INSERT INTO tUsers
```

```
        (uName, uFamily, uLogonName,
uPassword, IsActive)
```

```
VALUES
(@uName,@uFamily,@uLogonName,@uPassword,@IsActive)

RETURN
```

```
////////////////////////////////////////////////////////////////
```

```
ALTER PROCEDURE dbo.UsersSelect
```

```
AS
```

```
SELECT      uId, uName, uFamily, uLogonName, IsActive
FROM        tUsers

RETURN
```

```
////////////////////////////////////////////////////////////////
```

```
ALTER PROCEDURE dbo.UsersUpdate
```

```
@uId bigint ,
@uName nvarchar(50),
@uFamily nvarchar(50),
@uLogonName nvarchar(50),
@IsActive int
```

```
AS
```

```
UPDATE      tUsers

SET          uName = @uName, uFamily = @uFamily,
uLogonName = @uLogonName, IsActive = @IsActive

WHERE       (uId = @uId)

RETURN
```

```
////////////////////////////////////////////////////////////////
```

```
ALTER PROCEDURE dbo.UsesSelect
```

```
AS
```

```
SELECT      uId, uName, uFamily, uLogonName, IsActive  
FROM        tUsers  
RETURN
```

حال برگردید محیط صفحه را مطابق شکل زیر تنظیم نمایید:

<input type="text"/>	*	نام کاربر :
<input type="text"/>	*	نام خانوادگی کاربر:
<input type="text"/>	*	کلمه کاربری :
<input type="text"/>	*	کلمه عبور :
<input type="text"/>	*	نابود کلمه عبور :
<input type="text"/>	*	نابود کلمه عبور را نابودنمایید
		وضعیت کاربر : <input type="radio"/> فعال <input type="radio"/> غیر فعال
		ثبت

کدهای این تنظیمات نیز به شرح زیر است ولی به صورت Design نیز می توانید اینها را ایجاد کنید.

```
<div dir="rtl">  
    <table border="1">  
    <tr>  
        <td width="10%" >  
            نام کاربر :  
  
        </td>  
    <td >
```

```
<asp:TextBox ID="TextBox1" runat="server"
Width="279px" AutoCompleteType="Search"></asp:TextBox>
```

```
<asp:RequiredFieldValidator
ValidationGroup="ASRDA" ID="RequiredFieldValidator1"
runat="server" ErrorMessage="*"
ControlToValidate="TextBox1"></asp:RequiredFieldValidator>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td >
```

نام خانوادگی کاربر:

```
</td>
```

```
<td>
```

```
<asp:TextBox ID="TextBox2" runat="server"
Width="279px"></asp:TextBox>
```

```
<asp:RequiredFieldValidator
ValidationGroup="ASRDA" ID="RequiredFieldValidator2"
runat="server" ErrorMessage="*"
ControlToValidate="TextBox2"></asp:RequiredFieldValidator>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td >
```

کلمه کاربری:

```
</td>
```

```
<td>
```

```
<asp:TextBox ID="TextBox3" runat="server"
Width="279px"></asp:TextBox>
```

```
<asp:RequiredFieldValidator
ValidationGroup="ASRDA" ID="RequiredFieldValidator3"
runat="server" ErrorMessage="*"
ControlToValidate="TextBox3"></asp:RequiredFieldValidator>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td >
```

کلمه عبور:

```
</td>
```

```
<td>
```

```
<asp:TextBox ID="TextBox4" runat="server"
Width="279px"></asp:TextBox>
```

```
<asp:RequiredFieldValidator
ValidationGroup="ASRDA" ID="RequiredFieldValidator4"
runat="server" ErrorMessage="*"
ControlToValidate="TextBox4"></asp:RequiredFieldValidator>
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td >
```

تایید کلمه عبور:

```
</td>
```

```
<td>
```



```

        <asp:TextBox ID="TextBox5" runat="server"
Width="279px"></asp:TextBox>

        <asp:RequiredFieldValidator
ValidationGroup="ASRDA" ID="RequiredFieldValidator5"

        runat="server" ErrorMessage="*"
ControlToValidate="TextBox5"></asp:RequiredFieldValidator>

        <asp:CompareValidator
ValidationGroup="ASRDA" ControlToValidate="TextBox5"
ControlToCompare="TextBox4"

        ID="CompareValidator1" runat="server"
ErrorMessage="تاییدنمایید را عبور کلمه"></asp:CompareValidator>

    </td>
</tr>
<tr>
    <td >

        وضعیت کاربر:

    </td>
    <td>

        <asp:RadioButtonList ID="RadioButtonList1"
runat="server" RepeatDirection="Horizontal">

            <asp:ListItem Selected="True"
Value="1">فعال</asp:ListItem>

            <asp:ListItem Value="0">غیر
فعال</asp:ListItem>

        </asp:RadioButtonList>

    </td>
</tr>

```



```

        <RowStyle ForeColor="#000066" />
        <Columns>
نام "
        <asp:TemplateField HeaderText="
            SortExpression="uName">
            <ItemTemplate>
                <asp:Label ID="lbluName"
runat="server" Text='<%#Eval("uName")%>'></asp:Label>
            </ItemTemplate>
            <EditItemTemplate>
                <asp:TextBox ID="txtName"
runat="server" Text='<%#Bind("uName")%>'></asp:TextBox>
            </EditItemTemplate>
        </asp:TemplateField>
نام
        <asp:TemplateField HeaderText="
            خانوادگی">
            <ItemTemplate>
                <asp:Label ID="lbluFamily"
runat="server" Text='<%#Eval("uFamily")%>'></asp:Label>
            </ItemTemplate>
            <EditItemTemplate>
                <asp:TextBox ID="txtFamily"
runat="server" Text='<%#Bind("uFamily")%>'></asp:TextBox>
            </EditItemTemplate>
        </asp:TemplateField>
نام
        <asp:TemplateField HeaderText="
            کاربری">
            <ItemTemplate>

```

```

                <asp:Label
                    ID="lbluLogonName" runat="server"
Text='<%#Eval("uLogonName")%>'></asp:Label>
            </ItemTemplate>
        <EditItemTemplate>
            <asp:TextBox
                ID="txtLogonName" runat="server"
Text='<%#Bind("uLogonName")%>'></asp:TextBox>
        </EditItemTemplate>
    </asp:TemplateField>
    <asp:TemplateField
        >وضعیتHeaderText="
        <ItemTemplate>
            <asp:Label ID="lbluIsActive"
                runat="server"
Text='<%#GetStatusTitle(Eval("IsActive").ToString())%>'></asp:Label>
        </ItemTemplate>
        <EditItemTemplate>
            <asp:RadioButtonList
ID="RadioButtonList2" SelectedValue='<%#Bind("IsActive")%>'
                runat="server"
                RepeatDirection="Horizontal">
                <asp:ListItem
                    </asp:ListItem>فعالValue="1">
                <asp:ListItem
                    </asp:ListItem>فعال غيرValue="0">
            </asp:RadioButtonList>
        </EditItemTemplate>

```

```

</asp:TemplateField>

        <asp:CommandField
            ShowEditButton="True" />

        <asp:CommandField
            ShowDeleteButton="True" />

    </Columns>

    <FooterStyle BackColor="White"
        ForeColor="#000066" />

    <PagerStyle BackColor="White"
        ForeColor="#000066" HorizontalAlign="Left" />

    <SelectedRowStyle BackColor="#669999"
        Font-Bold="True" ForeColor="White" />

    <HeaderStyle BackColor="#006699" Font-
        Bold="True" ForeColor="White" />

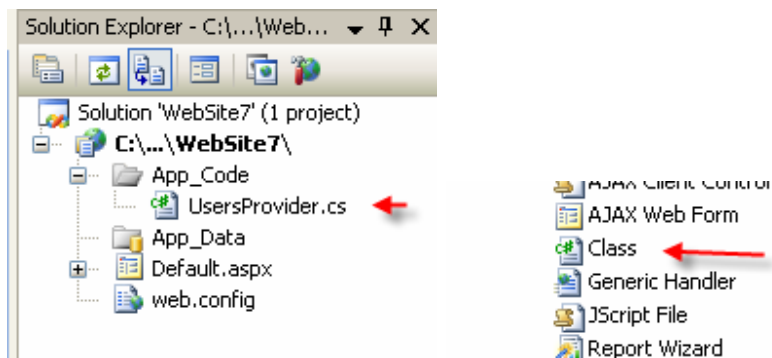
</asp:GridView>

```

asp:GridView#GridView1					ثبت
نام	نام خانوادگی	نام کاربری	وضعیت	Delete	Edit
Databound	Databound	Databound	Databound	Delete	Edit
Databound	Databound	Databound	Databound	Delete	Edit
Databound	Databound	Databound	Databound	Delete	Edit
Databound	Databound	Databound	Databound	Delete	Edit

1

حال یک کلاس با نام UsersProvider.cs ایجاد کنید



و کدهای زیر را درون آن بنویسید

```
using System;
using System.Data;
using System.Configuration;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Data.SqlClient;

public class UsersProvider
{
    SqlConnection MConnection = new SqlConnection();
    SqlCommand MCommand = new SqlCommand();
    SqlDataAdapter MAdapter = new SqlDataAdapter();
    DataTable MDatatable = new DataTable();

    public UsersProvider()
    {
```

```

}

private void OpenConnection(string CommnadText)
{
    MConnection.ConnectionString = "Data
Source=.\SQLEXPRESS;AttachDbFilename=" + @"C:\Documents and
Settings\Administrator\Desktop>LoginTest(control in
grid)\Login.mdf" + ";Integrated Security=True;Connect
Timeout=30;User Instance=True";

    MConnection.Open();

    MCommand.Connection = MConnection;

    MCommand.CommandType = CommandType.StoredProcedure;

    MCommand.CommandText = CommnadText;
}

private void CloseConnection()
{
    MConnection.Close();
}

public void UsersInsert(string uName, string uFamily, string
uLogonName, string uPassword, int IsActive)
{
    OpenConnection("UsersInsert");

    MCommand.Parameters.Add("@uName",
SqlDbType.NVarChar).Value = uName;

    MCommand.Parameters.Add("@uFamily",
SqlDbType.NVarChar).Value = uFamily;
}

```

```

        MCommand.Parameters.Add("@uLogonName",
SqlDbType.NVarChar).Value = uLogonName;

        MCommand.Parameters.Add("@uPassword",
SqlDbType.NVarChar).Value = uPassword;

        MCommand.Parameters.Add("@IsActive",
SqlDbType.Int).Value = IsActive;

        MCommand.ExecuteNonQuery();

        CloseConnection();

    }

    public void UsersUpdate(long uId, string uName, string
uFamily, string uLogonName, string uPassword, int IsActive)
    {

        OpenConnection("UsersUpdate");

        MCommand.Parameters.Add("@uId", SqlDbType.BigInt).Value
= uId;

        MCommand.Parameters.Add("@uName",
SqlDbType.NVarChar).Value = uName;

        MCommand.Parameters.Add("@uFamily",
SqlDbType.NVarChar).Value = uFamily;

        MCommand.Parameters.Add("@uLogonName",
SqlDbType.NVarChar).Value = uLogonName;

        MCommand.Parameters.Add("@IsActive",
SqlDbType.Int).Value = IsActive;

        MCommand.ExecuteNonQuery();

        CloseConnection();

    }

    public void UsersDelete(long uId)
    {

        OpenConnection("UsersDelete");

```



```

        MCommand.Parameters.Add("@uId", SqlDbType.BigInt).Value
= uId;

        MCommand.ExecuteNonQuery();

        CloseConnection();
    }

    public DataTable UsersSelect()
    {
        OpenConnection("UsersSelect");
        MAdapter.SelectCommand = MCommand;
        MAdapter.Fill(MDatatable);
        CloseConnection();
        return MDatatable;
    }
}

```

از این کلاس برای ارتباط با دیتابیس استفاده می شود.

بعد یک **ObjectDataSource** بر روی صفحه قرار دهید کدهای آنرا به شرح زیر تغییر دهید:

```

<asp:ObjectDataSource ID="ObjectDataSource1" runat="server"
    DeleteMethod="UsersDelete"
    InsertMethod="UsersInsert"
    SelectMethod="UsersSelect" TypeName="UsersProvider"
    UpdateMethod="UsersUpdate">
    <DeleteParameters>
        <asp:Parameter Name="uId"
            Type="Int64" />

```

```
        </DeleteParameters>

        <UpdateParameters>

            <asp:Parameter Name="uId"
                Type="Int64" />

            <asp:Parameter Name="uName"
                Type="String" />

            <asp:Parameter Name="uFamily"
                Type="String" />

        <asp:Parameter Name="uLogonName"
            Type="String" />

        <asp:Parameter Name="uPassword"
            Type="String" />

        <asp:Parameter Name="IsActive"
            Type="Int32" />

        </UpdateParameters>

        <InsertParameters>

            <asp:Parameter Name="uName"
                Type="String" />

            <asp:Parameter Name="uFamily"
                Type="String" />

        <asp:Parameter Name="uLogonName"
            Type="String" />

        <asp:Parameter Name="uPassword"
            Type="String" />

        <asp:Parameter Name="IsActive"
            Type="Int32" />

        </InsertParameters>

    </asp:ObjectDataSource>
```

در قسمت آخر نیز کدهای دکمه ثبت و یک تابع را بنویسید

```
protected void Button1_Click(object sender, EventArgs e)
{
    ObjectDataSource1.InsertParameters["uName"].DefaultValue
        = TextBox1.Text;

    ObjectDataSource1.InsertParameters["uFamily"].DefaultValue =
        TextBox2.Text;

    ObjectDataSource1.InsertParameters["uLogonName"].DefaultValue =
        TextBox3.Text;

    ObjectDataSource1.InsertParameters["uPassword"].DefaultValue =
        TextBox4.Text;

    ObjectDataSource1.InsertParameters["IsActive"].DefaultValue =
        RadioButtonList1.SelectedValue;

    ObjectDataSource1.Insert();
}

public string GetStatusTitle(string StatusId)
{
    return (StatusId == "0" ? "
    غیر فعال" : "فعال");
}
```

حال برنامه را اجرا کنید و لذت ببرید از برنامه :



نکته مهمی که باید به آن اشاره کنم امنیت کوکی ها است یعنی شما نباید اطلاعات خود را بدون رمزنگاری در کوکی ذخیره کنید چون کاربر می تواند آن را باز کرده و به دلخواه اطلاعات شما را تغییر دهد برای رمز نگاری می توانید از الگوریتمهایی که در فصلهای قبل توضیح دادم استفاده کنید مثل الگوریتم رایندال!!!

مثال : یک برنامه وب ایجاد کنید و یک دکمه در صفحه اول و یک تکست باکس در صفحه دوم قرار دهید و کدهای زیر را برای دکمه صفحه اول و **load** صفحه دوم بنویسید.می خواهیم اطلاعاتی را در سمت کاربر ذخیره کنم و در جایی دیگر درخواستی که از سمت کاربر آمده توی کوکی های آن می گردم و اطلاعات خود را برمی دارم.

```
protected void Button1_Click(object sender, EventArgs e)
{
    HttpCookie _HCK = new HttpCookie("num1", "my name is javad");
    Response.Cookies.Add(_HCK);
    Response.Redirect("Default2.aspx");
}
```

////////////////////////////////////

```
protected void Page_Load(object sender, EventArgs e)
{
    if (Request.Cookies["num1"] != null)
    {
        TextBox1.Text = Request.Cookies["num1"].Value;
    }
}
```

برای تعیین زمان پاک شدن کوکی ( مثلا بعد از ۱۰ دقیقه پاک شود) می توانید به این شکل عمل کنید

```
protected void Button1_Click(object sender, EventArgs e)
{
    HttpCookie _HCK = new HttpCookie("num1", "my name is javad");
    _HCK.Expires = DateTime.Now.AddMinutes(10);
    Response.Cookies.Add(_HCK);
    Response.Redirect("Default2.aspx");
}
```

حال زمان کوکی را زمان حال قرار دهید و برنامه را اجرا کنید مشاهده خواهید کرد که هیچ اطلاعاتی نمایش داده نمی شود چون زمان انقضا کوکی از بین رفته است.

```
_HCK.Expires = DateTime.Now;
```

مراجع :

MSDN-١

Apress.LINQ.for.Visual.C.Sharp.2008.Jun.2008-٢

Microsoft Sql Server 2005 Programming For Dummies Apr 2007-٣

Professional C# 2008 (Wrox - Mar 2008-٤

-٥

Prentice.Hall.Core.Internet.Application.Development.with.ASP.NET.2.0.Feb  
.2007

Wrox.Professional.ASP.NET3.5 2008-٦