بسم الله الرحمن الرحیم

برنامه‌سازی ۳

رشتهٔ کامپیوتر

گروه تحصیلی کامپیوتر

زمینهٔ خدمات

شاخهٔ آموزش فنی وحرفه‌ای

۱۳۹۵

همکاران محترم و دانش‌آموزان عزیز :

پیشنهادات و نظرات خود را دربارهٔ محتوای این کتاب بـه نشانی تهران ـ صندوق پستی شمارهٔ ۴۸۷۴/۱۵ دفتر تألیف‌کتاب‌های درسی فنی و حرفه‌ای و کاردانش، ارسال فرمایند.

پیام‌نگار (ایمیل) **info@tvoccd.sch.ir**

وب‌گاه (وب‌سایت) **www.tvoccd.sch.ir**

وزارت آموزش و پرورش
سازمان پژوهش و برنامه‌ریزی آموزشی

برنامه‌ریزی محتوا و نظارت بر تألیف : **دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش**

نام کتاب : **برنامه‌سازی ۳ ـ ۴۵۱/۵**

مؤلف : **مجید کربلایی**

اعضای کمیسیون تخصصی : **محمدرضا یمقانی، مهیار پازوکی، نیلوفر بزرگ‌نیا، محمدرضا شکرریز، مهناز کارکن، سید سعید رضا سعادت‌یزدی، پردیس پیرایش، زهرا عسگری رکن‌آبادی و حسین صفوی**

آماده‌سازی و نظارت بر چاپ و توزیع : **ادارهٔ کل نظارت بر نشر و توزیع مواد آموزشی**

تهران : خیابان ایرانشهر شمالی ـ ساختمان شمارهٔ ۴ آموزش و پرورش (شهید موسوی)

تلفن : ۹ـ ۸۸۸۳۱۱۶۱، دورنگار : ۸۸۳۰۹۲۶۶، کد پستی :۱۵۸۴۷۴۷۳۵۹،

وب سایت : www.chap.sch.ir

مدیر امور فنی و چاپ : **لیدا نیک‌روش**

طراح جلد : **مریم کیوان**

صفحه‌آرا : **زهره بهشتی‌شیرازی**

حروفچین : **کبری اجابتی**

مصحح : **پری ایلخانی‌زاده، شهلا دالایی**

امور آماده‌سازی خبر : **زینت بهشتی‌شیرازی**

امور فنی رایانه‌ای : **ناهید خیام‌باشی، راحله زادفتح‌اله**

ناشر : **شرکت چاپ و نشر کتاب‌های درسی ایران : تهران ـ کیلومتر ۱۷ جادّهٔ مخصوص کرج ـ خیابان ۶۱ (دارو پخش)**

تلفن : ۵ ـ ۴۴۹۸۵۱۶۱، دورنگار : ۴۴۹۸۵۱۶۰، صندوق پستی : ۱۳۹ـ ۳۷۵۱۵

چاپخانه : **شرکت چاپ و نشر کتاب‌های درسی ایران «سهامی خاص»**

سال انتشار و نوبت چاپ : **چاپ دوم ۱۳۹۵**

شابک ۱-۲۴۴۱-۰۵-۹۶۴-۹۷۸ ISBN 978-964-05-2441-1

من در اینجا به جوانان عزیز کشورمان، به این سرمایه‌ها و ذخیره‌های عظیم الهی و به این گل‌های معطر و نوشکفتهٔ جهان اسلام سفارش می‌کنم که قدر و قیمت لحظات شیرین زندگی خود را بدانید و خودتان را برای یک مبارزهٔ علمی و عملی بزرگ تا رسیدن به اهداف عالی انقلاب اسلامی آماده کنید.

امام خمینی «ره»

فهرست

مقدمه

کتاب برنامه‌سازی ۳، در ادامه کتاب‌های برنامه‌سازی ۱ و ۲ برای آشنایی هنرجویان فنی‌وحرفه‌ای رشته کامپیوتر، با زبان برنامه‌سازی C# تألیف شده است. در این کتاب مانند دو کتاب قبلی سعی شده است که زبان برنامه‌نویسی C# با زبانی ساده و قابل فهم برای هنرجویان بیان شود. در ابتدای هر فصل، هدف‌های رفتاری ذکر شده است و در انتهای هر فصل خودآزمایی، تمرینات برنامه‌نویسی و جدول واژگان انگلیسی نیز قرار دارد که هنرجویان عزیز باید آنها را انجام دهند تا به اهداف رفتاری ذکر شده در ابتدای هر فصل برسند. بیشتر مطالب درسی این کتاب در بخش کار در کارگاه در هر فصل توضیح داده شده است که باید به‌صورت عملی انجام شود. مثال‌های ارائه شده در بخش کار در کارگاه، قدم به قدم هنرجویان را برای ساخت یک برنامه کاربردی جلو می‌برد. آنچه که از هنرجویان انتظار می‌رود انجام دقیق مثال‌های بخش کار در کارگاه و حل تمرینات و پاسخ به خودآزمایی‌ها است که با انجام آن، زمینهٔ لازم برای یادگیری مطالب کتاب فراهم می‌شود.

در فصل اول کتاب، استفاده از چند کنترل پیشرفته بیان شده است. از آنجا که به علت تعدد کنترل‌ها، امکان توضیح همه کنترل‌ها در این کتاب میسر نیست، هنرجویان عزیز، خود باید کنترل‌های موردنیاز در یک برنامه، نظیر کنترل ComboBox و ListBox را از طریق جعبه ابزارها شناسایی و با استفاده از منابع در اختیار نظیر اینترنت، به‌کار گیرند. در فصل دوم رویدادهای ماوس و صفحه کلید با ذکر مثال توضیح داده شده است. در فصل سوم منو و کار با ابزارهای آن و در فصل چهارم روش تعریف کلاس و ایجاد شیء بیان شده است. در فصل پنجم در مورد روش ایجاد و خواندن فایل‌های متنی توضیح داده شده است. و بالاخره در فصل ۶ که فصل پایانی کتاب است در مورد روش اتصال

برنامه به پایگاه داده در قالب پروژه توضیح داده شده است. برای ساخت پایگاه داده از ابزارهای قدرتمند ویژوال استودیو اکسپرس استفاده شده است. نوع پایگاه داده مطرح شده در فصل آخر Microsoft SQL Server Compact Edition است. هنرآموزان گرامی و هنرجویان عزیز توجه داشته باشند که پروژه‌های مطرح شده در کتاب، اختیاری نیستند و مانند مطالب دیگر کتاب باید در کلاس، تدریس و به وسیلهٔ هنرجویان با دقت انجام شوند زیرا در حین اجرای مرحله به مرحلهٔ پروژه‌ها، مطالب درسی بیان شده است. خواهشمندم در ارایه مثال‌ها و تمرینات جانبی که کتاب به آن نیازمند است، جنبه کاربردی و معنی‌دار بودن آنها را درنظر داشته باشند. هنرجویان عزیز نیز تلاش کنند که این پروژه‌ها را گسترش دهند، تا بتوانند از آنها در یک محیط واقعی استفاده نمایند.

هنرآموزان عزیز لطفاً از بخش فعالیت برای طراحی آزمون‌های پیشرفت تحصیلی استفاده نکنند. در بخش فعالیت در پایان هر فصل یک پروژه به‌ترتیب و متناسب با فصول کتاب تکمیل می‌شود. این پروژه برای فراگیری بهتر، پروژه به شکل اختیاری است ولی پیشنهاد می‌شود کارگاهی تکمیل و انجام شود.

در پایان از تمام عزیزانی که در تألیف این کتاب به اینجانب کمک کردند، تشکر و قدردانی می‌نمایم. با وجود دقتی که در تألیف این کتاب صورت گرفته است، کتاب عاری از اشتباه نیست، اشکالات و پیشنهادات شما باعث بهبود کیفیت کتاب خواهد بود لذا مواردی که به‌نظر شما می‌رسد را دریغ نفرمایید و به اطلاع ما برسانید.

فصل اول

آشنایی با چند کنترل جدید

در کتاب برنامه سازی ۲، به طور خاص روی پروژه‌های ویندوزی کار کردیم و با آنها واسط‌های کاربر گرافیکی تولید نمودیم. کنترل‌های زیادی هستند که می‌توانند این واسط‌های گرافیکی را از منظر کاربران محبوب‌تر و دلچسب‌تر نمایند. در این فصل با چند کنترل دیگر آشنا می‌شویم. در این‌گونه پروژه‌ها کنترل‌های ورود داده همیشه پرکاربرد و با اهمیت هستند. بنابراین در این فصل تمرین بیشتری با چند کنترل مهم ورود داده انجام می‌دهیم و با کنترل‌های جدیدی آشنا می‌شویم. در مثال‌های این فصل با فرم‌هایی کار می‌کنیم که داده‌ها را برای مقاصد مختلف از کاربر می‌گیرند.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ فرم‌های ورود اطلاعات متفاوت تولید نماید.

۲ــ تفاوت و کاربرد کنترل دکمه انتخاب و کادر انتخاب را توضیح دهد و آن‌ها را در برنامه خود به کار بندد.

۳ــ کادر محاوره‌ای پیام را با دکمه‌های متنوع ایجاد نماید.

۴ــ کنترل NumericUpDown را در برنامه‌های خود استفاده کند.

۱ــ۱ــ ایجاد فرم ورود اطلاعات فردی

مثال ۱ــ۱ــ برنامه‌ای بنویسید که بتواند یک فرم ورود اطلاعات مانند شکل ۱ــ۱ ایجاد نماید.

شکل ۱ــ۱ــ فرم ورود اطلاعات

الگوریتم یا روش انجام کار

۱ــ ایجاد یک پروژه ویندوزی: در برنامه VS یک پروژه جدید از نوع Windows Form Application با نام پروژه Entry Information در مسیر مشخصی بسازید.

۲ــ درج کنترل‌ها: ساخت یک فرم ورود اطلاعات مانند شکل ۱ــ۱ ساده است و پیچیدگی خاصی ندارد. تنها کنترلی که در فرم وجود دارد، که قبلاً با آن برخورد نکرده‌ایم، قسمت انتخاب جنسیت است که می‌توان با کلیک بر روی دایره‌ها (دکمه‌ها)، جنسیت مرد یا زن را انتخاب کرد. این دکمه‌ها، به دکمه رادیویی[۱] معروف هستند. چرا؟

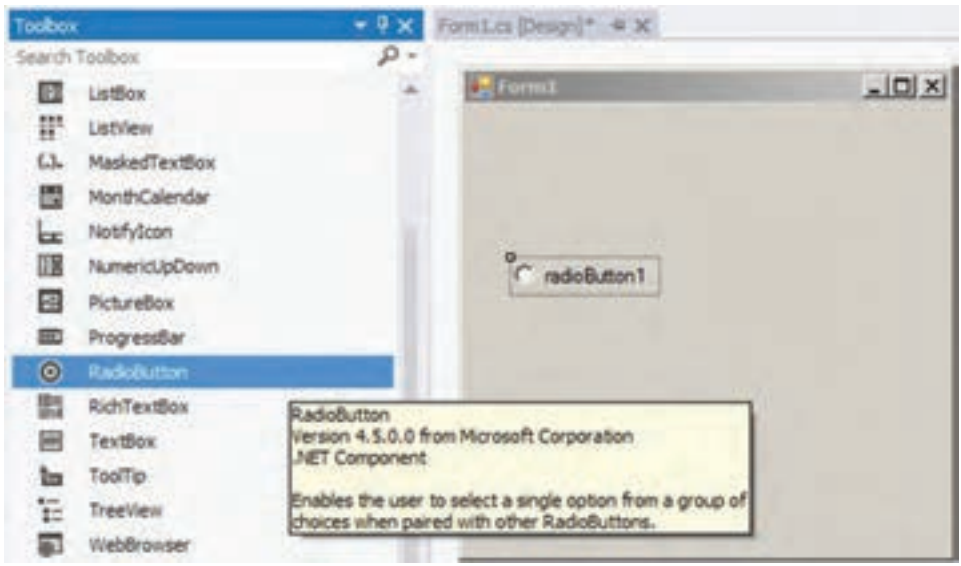
---

۱ ــ Radio Button

نکته

در کنترل دکمه رادیویی، فقط یک گزینه حق انتخاب دارد.

در روی فرم‌ها نیز معمولاً برای انتخاب یک مورد از بین دو یا چند مورد، مانند انتخاب جنسیت (مرد[1]، زن[2])، انتخاب رنگ زمینه (سبز، آبی، قرمز) و یا انتخاب نحوه پرداخت (نقدی[3]، کارت[4])، چک از دکمه‌های رادیویی استفاده می‌کنیم. مثلاً برای انتخاب جنسیت نیاز به دو دکمه رادیویی است. برای اضافه کردن کنترل دکمه رادیویی، به جعبه ابزار Common Controls مراجعه و کنترل RadioButton را به فرم اضافه می‌کنیم. (شکل ۲ــ۱)



شکل ۲ــ۱

۱ ــ Male

۲ ــ Female

۳ ــ Cash

۴ ــ Debit Card

۳ــ تعیین ویژگی‌های کنترل‌ها

جدول ۱ــ۱ــ ویژگی‌های برچسب

Label						
ویژگی	مقدار	مقدار	مقدار	مقدار	مقدار	مقدار
Name	nameText	familyText	fatherText	telText	mobileText	emailText
Text	نام	نام خانوادگی	نام پدر	تلفن ثابت	تلفن همراه	پست الکترونیکی

جدول ۲ــ۱ ویژگی‌های کادر انتخاب

CheckBox	
ویژگی	مقدار
Name	acceptCheck
Checked	False
RightToLeft	Yes

جدول ۳ــ۱ــ ویژگی‌های دکمهٔ انتخاب

RadioButton		
ویژگی	مقدار	مقدار
Name	maleRadio	femaleRadio
Checked	False	False
RightToLeft	Yes	Yes

برای دکمه‌های رادیویی و دکمه‌های انتخاب فارسی، ویژگی RightToLeft را Yes انتخاب می‌کنیم تا علامت دایره و مربع در سمت راست نوشته قرار گیرد.

نکته

در صورتی که بخواهید دو دسته دکمه رادیویی مستقل از یکدیگر، روی فرم داشته باشید، باید هر دسته از دکمه‌ها را در داخل یک قاب مانند (GroupBox) قرار دهید.

سؤال: برای اینکه جنسیت مرد، انتخابِ پیش فرض باشد، چه ویژگی‌ای مقداردهی می‌شود؟

۴ــ نوشتن کد برای دکمه «ذخیره» : متد EH را برای دکمه ذخیره با رعایت شرایط زیر، می‌نویسیم :

الف) تمام اطلاعات خواسته شده که با علامت ستاره قرمز رنگ مشخص شده است اجباری است و کاربر باید وارد کند (کادرهای متنی نباید خالی باشد).

ب) با کلیک بر روی دکمه «ذخیره»، علاوه بر رعایت بند الف، باید کادر تیک مربوط به «قبول شرایط عضویت در سیستم» نیز تیک خورده باشد. اگر هر دو شرط برقرار بود، پیام «اطلاعات عضو جدید در سیستم ثبت شد» نشان داده شود و در غیر این صورت پیام خطای مناسبی ظاهر شود.

با استفاده از دستور if خالی نبودن کادرهای متنی و علامت دار بودن checkbox و انتخاب شدن یکی از RadioButtonها بررسی می‌شود.

```
if (name Text. Text != " " && family Text. Text != " "
        && father Text. Text != " " && telText. Text != " "
                && mobileText.Text != " " && emailText. Text != " "
                    && acceptCheck. Checked
                        && (maleRadio.Checked || femaleRadio.Checked))
{


}
```

بنابراین متد EH مربوط به دکمه «ذخیره» می‌تواند چنین باشد :

```
Private void saveButton_ Click(object sender, EventArgs e)
{
    if (name Text. Text != " " && family Text. Text != " "
        && father Text. Text != " " && telText. Text != " "
            && mobileText.Text != " " && emailText. Text != " "
                && acceptCheck. Checked
                    && (maleRadio.Checked || femaleRadio.Checked))
    {
        string message = "اطلاعات عضو جدید در سیستم ثبت شد" "+" "\n نام : "+nameText.Text
                    +"\n نام خانوادگی : "+familyText.Text+"\n نام پدر : "+fatherText.Text
                    +"\n تلفن ثابت : "+telText.Text+"\n تلفن همراه : "+mobileText.Text
                    +"\n پست الکترونیک : "+emailText.Text+"\n جنسیت : ";
        if (maleRadio. Checked)
            message + = "مرد";
        else
            message + = "زن";
        MessageBox. Show (message, "ورود اطلاعات کاربر" );
    }
    else
    {
        MessageBox. Show ("اطلاعات باید به‌طور کامل وارد شود" , "ورود اطلاعات کاربر");
    }
}
```

۵ــ نوشتن کد برای دکمه «انصراف»: متد EH را برای دکمه انصراف با رعایت شرایط زیر، می‌نویسیم :

الف) با کلیک بر روی دکمه انصراف، یک پیام مبنی بر «اطمینان از انصراف» ظاهر شود (شکل ۳ــ۱).

شکل ۳ــ۱ــ پیام هشدار خروج از برنامه

ب) اگر کاربر دکمه YES را انتخاب کرد فرم ورود اطلاعات بسته شود و اگر دکمه NO را انتخاب کرد، اتفاقی نیفتد.

بنابراین، متد EH مربوط به دکمه "انصراف" می‌تواند چنین باشد :

```
Private void cancelButton_ Click(object sender, EventArgs e)
{
    string message = "آیا قصد دارید از برنامه خارج شوید؟" ;
    string caption = "هشدار" ;
    DialogResult result = MessageBox. Show (message, caption,
MessageBox Buttons. YesNo);
    if (result = = Dialog Result. Yes)
        this. Close();
}
```

از نوع داده شمارشی DialogResult برای بررسی مقدار برگشتی دکمه فشرده شده در کادر محاوره‌ای استفاده می‌شود. مقادیر برگشتی DialogResult را در جدول ۴ــ۱ ببینید.

جدول ۴ــ۱ــ مقادیر DialogResult

مقدار برگشتی	شرح
None	از کادر محاوره‌ای چیزی برنگشته است.
OK	دکمه ok فشرده شده است.
Cancel	دکمه Cancel فشرده شده است.
Abort	دکمه Abort فشرده شده است.
Retry	دکمه Retry فشرده شده است.
Ignore	دکمه Ignore فشرده شده است.
Yes	دکمه Yes فشرده شده است.
No	دکمه No فشرده شده است.

کلمه کلیدی this به نمونه‌ای از کلاس Form۱ اشاره می‌کند و منظور کلاسی است که رویداد برای آن نوشته شده است و متد Close برای بستن فرم استفاده می‌شود.

توسعه و بهبود برنامه

تمرین : در رویداد دکمه ذخیره اطلاعات با توجه به خالی بودن هر یک از کادرهای متنی پیامی متناسب با آن نمایش داده شود. برای نمونه در صورت خالی بودن کادر متنی nameText پیام «لطفا نام را وارد کنید» نمایش داده شود.

۲ــ۱ــ کنترل عددی افزایشی ــ کاهشی

علاوه بر کادر متنی، می‌توانید از کنترل عددی افزایشی ــ کاهشی (NumericUpDown) برای دریافت داده‌های عددی، در یک محدوده مشخص، استفاده نمایید. در مثال زیر کنترل مذکور در قالب یک برنامه به نام سنجش مهارت عمل جمع، به‌کار رفته است.

کار در کارگاه ۲: سنجش مهارت عمل جمع

مثال ۲ــ۱ : برنامه‌ای بنویسید که برای سنجش مهارت عمل جمع یک دانش‌آموز، مورد استفاده قرار گیرد. در این برنامه ابتدا فرمی‌مانند شکل ۴ــ۱ ظاهر می‌شود.

شکل ۴ــ۱

کاربر با کلیک بر روی دکمه «شروع»، باید به سؤالی که ظاهر می‌شود، در مدت زمان محدودی، پاسخ دهد. اعدادی که باید جمع زده شوند در محدوده یک تا ۵۰ به‌طور تصادفی انتخاب می‌گردند.

مثلاً در شکل ۵ــ۱ اعداد ۴۸ و ۳ باید با یکدیگر جمع زده شوند.

شکل ۵ ــ۱ــ فرم برنامه سنجش مهارت جمع

کاربر می‌تواند عدد مورد نظر خود را در قسمت سفید رنگ تایپ کند و یا با استفاده از فلش‌های بالا و پایین، قادر است پاسخ خود را وارد کند. اگر کاربر بتواند در مدت تعیین شده پاسخ صحیح دهد، پیامی‌مانند شکل ۶ــ۱ ظاهر می‌شود :

شکل ۶ ــ۱ــ نمایش پیام برای پاسخ درست

چنانچه کاربر نتواند در مدت زمان تعیین شده پاسخ صحیح دهد، پیامی‌مانند شکل ۷ــ۱ ظاهر می‌شود :

شکل ۷ ــ۱ــ نمایش پیام برای پاسخ اشتباه

پس از پایان مدت زمان تعیین شده و نمایش پیام‌های مربوطه، کاربر می‌تواند با کلیک بر روی دکمه «شروع» سؤال دیگری را مشاهده کند و در مدت زمان تعیین شده پاسخ دهد.

الگوریتم یا روش انجام کار : مراحل زیر را برای انجام این مثال دنبال می‌کنیم.

۱ــ ایجاد یک پروژه ویندوزی : در برنامه VS یک پروژه جدید از نوع Windows Form Application با نام پروژه SimpleQuiz در مسیر مشخصی بسازید.

۲ــ تعیین ویژگی‌های فرم : یک فرم ورود اطلاعات مانند شکل ۷ــ۱ ساده است و پیچیدگی خاصی ندارد. ویژگی‌های فرم را مانند جدول زیر تعیین کنید :

جدول ۵ــ۱ــ ویژگی‌های فرم

Form	
ویژگی	مقدار
Name	Form1
Text	سنجش مهارت جمع

۳ــ ایجاد کنترل‌های برچسب و تعیین ویژگی‌های آنها : در این پروژه نیاز به شش کنترل برچسب داریم.

ابتدا در روی فرم، دو برچسب برای نمایش دو عددی که باید جمع زده شود، قرار دهید.

جدول ۶ــ۱ــ ویژگی‌های کنترل برچسب اول

Label		
ویژگی		مقدار
Name		rightLabel
Text		?
Font		Arial Size: 16 Bold
Location	X	120
	Y	110

جدول ۷ــ۱ــ ویژگی‌های کنترل برچسب دوم

Label		
ویژگی		مقدار
Name		leftLable
Text		?
Font		Arial Size: 16 Bold
Location	X	10
	Y	110

نکته

اگر به جدول بالا دقت کنید مشاهده می‌کنید که اغلب ویژگی‌ها در دو برچسب مانند یکدیگر است. در این‌گونه موارد، می‌توانید یک برچسب را در روی فرم قرار دهید و ویژگی‌های آن را مطابق جدول تنظیم کنید، سپس آن را انتخاب کرده و عمل کپی و الحاق روی فرم را انجام دهید. در انتهای کار، فقط لازم است برخی از ویژگی‌های آن را مطابق با جدول تغییر دهید. استفاده از این روش، سرعت ساخت فرم را افزایش می‌دهد.

سپس در روی فرم، دو برچسب دیگر، برای نمایش نمادهای + و = قرار دهید.

جدول ۸ــ۱ــ ویژگی‌های کنترل برچسب سوم

Label		
ویژگی		مقدار
Name		equalLabel
Text		=
Font		Arial Size: 16 Bold
Location	X	170
	Y	110

جدول ۹ــ۱ــ ویژگی‌های کنترل برچسب چهارم

Label		
ویژگی		مقدار
Name		sumLabel
Text		+
Font		Arial Size: 16 Bold
Location	X	70
	Y	110

در روی فرم، دو برچسب دیگر برای نمایش پیام مربوط به زمان باقیمانده و اندازه ثانیه قرار دهید.

جدول ۱۰ــ۱ــ ویژگی‌های کنترل برچسب پنجم

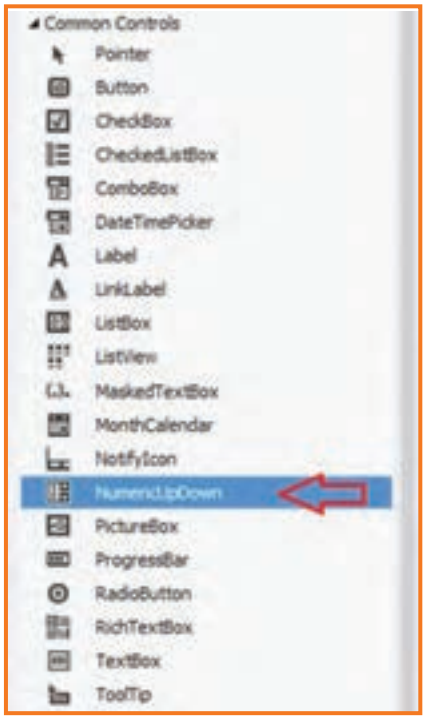
Label		
ویژگی		مقدار
Name		timeLabel
Text		
Visible		False
Font		Arial Size: 16 Bold
RightToLeft		No
Location	X	60
	Y	25

جدول ۱۱ــ۱ــ ویژگی‌های کنترل برچسب ششم

Label		
ویژگی		مقدار
Name		messageLabel
Text		مدت زمان باقیمانده
Visible		False
Font		Arial Size: 16 Bold
RightToLeft		Yes
Location	X	120
	Y	30

۴ــ ایجاد کنترل عددی افزایشی ــ کاهشی:

برای این منظور به جعبه ابزار Common Controls رفته و ابزار NumericUpDown را انتخاب و روی فرم قرار دهید.



شکل ۸ــ۱

۵ــ تعیین ویژگی‌های کنترل عددی افزایشی ــ کاهشی: ویژگی‌های این کنترل را مطابق با جدول زیر تنظیم کنید.

جدول ۱۲ــ۱ــ ویژگی‌های کنترل عددی افزایشی ــ کاهشی

NumericUpDown	
ویژگی	مقدار
Name	numericUpDown1
Maximum	100
Minimum	0
Visible	False
Font	Arial Size: 16 Bold

حداقل و حداکثر مقداری که می‌توان در کنترل عددی وارد کرد، به وسیلهٔ دو ویژگی Minimum و Maximum معین می‌شود. با توجه به اینکه، دو عدد تصادفی بین ۰ تا ۵۰ ظاهر می‌شوند، بنابراین مجموع آنها، عددی بین ۰ تا ۱۰۰ است. از این‌رو مقدار ویژگی‌های Minimum و Maximum کنترل عددی را به ترتیب برابر صفر و ۱۰۰ مقداردهی می‌کنیم.

۶ــ ایجاد کنترل دکمه و تعیین ویژگی‌های آن: یک دکمه نیز به فرم اضافه کنید.

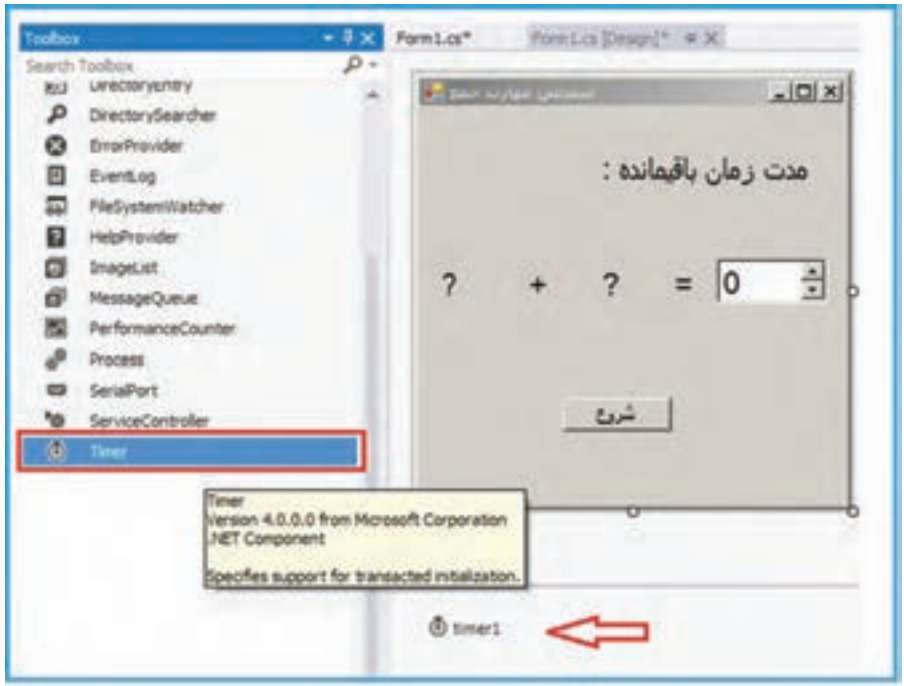
جدول ۱۳ــ۱ــ ویژگی‌های دکمه شروع

Button		
ویژگی		**مقدار**
Name		start
Text		شروع
Location	X	وسط صفحه قرار گیرد
	Y	200

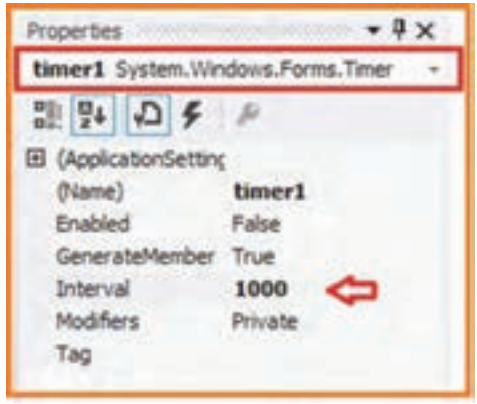
۷ــ ایجاد زمان‌سنج (Timer): با توجه به اینکه برای پاسخ به سؤال، محدودیت زمانی داریم و باید در ظرف مدت ۵ ثانیه به سؤال پاسخ داده شود، لذا از یک زمانی استفاده می‌کنیم. برای این منظور به سراغ پنجره جعبه ابزار رفته و در جعبه ابزار Component، یک شیء Timer را به برنامه اضافه کنید. توجه داشته باشید که این شیء در روی فرم نشان داده نمی‌شود بلکه در زیر فرم قرار می‌گیرد (شکل ۹ــ۱).

۸ــ تعیین ویژگی کنترل زمانی: با اضافه کردن شیء تایمر در برنامه، ویژگی‌های آن در پنجره Properties نشان داده می‌شود. ویژگی Interval را برابر ۱۰۰۰ قرار دهید تا هر ۱ ثانیه معادل ۱۰۰۰ میلی ثانیه، یک رویدادِ زمانی رخ دهد که در آن صورت متد EH مربوط به Timer به طور خودکار اجرا می‌شود (شکل ۱۰ــ۱).

سؤال: ویژگی Enabled برای زمان‌سنج به‌طور پیش فرض چه مقداری است؟



شکل ۹ ــ۱ــ اضافه کردن زمان‌سنج



شکل ۱۰ ــ۱ــ تعیین بازه زمانی زمان‌سنج

۹ــ *تعریف متغیرهای مورد نیاز برنامه :* قبل از تولید اعداد، دو متغیر برای نگهداری اعداد تولید شده تعریف می‌کنیم. این دو متغیر را در خط‌های ابتدایی کلاس Form۱ ایجاد می‌کنیم. همچنین در همین مکان یک متغیر با نام timeLeft برای نگهداری زمان باقیمانده تعریف می‌کنیم. با

این کار در همه متدها و رویدادهای کلاس می‌توانیم از این سه متغیر استفاده کنیم.

```
public partial class Form1 : Form
{
        int leftNumber;
        int rightNumber;
        int timeLeft;
```

۱۰ــ نوشتن رویداد Click برای دکمه : اکنون نوبت به نوشتن متد EH رویداد کلیک دکمه «شروع» می‌رسد. کارهایی که باید در این متد انجام شود عبارت‌اند از :

الف) تولید دو عدد تصادفی در محدوده یک تا پنجاه : برای تولید اعداد تصادفی صحیح، از کلاس Random استفاده می‌شود. برای استفاده از آن، کد زیر را درون دکمه شروع می‌نویسیم :

```
Random randomNumber = new Random();
```

اکنون اگر از دستور randomNumber.Next() استفاده کنیم یک عدد صحیح تصادفی خواهیم داشت. اما می‌خواهیم دامنه اعدادی که بدست می‌آوریم محدود باشد. یک راه این است که از متد Random.Next(maxValue) استفاده کنیم. این متد عددی بزرگ‌تر یا مساوی صفر و کوچک‌تر از maxValue به ما می‌دهد. توجه داشته باشید که خود عدد maxValue جزو محدوده نیست. بنابراین اگر از دستور randomNumber.Next(۵۰) استفاده کنیم عددی بین ۰ تا ۴۹ خواهیم داشت. پس با انتخاب حد بالایی ۵۱، به عددی بین ۱ تا ۵۰ دست پیدا می‌کنیم.

اما روش دیگر آن است که از متد Random.Next(minValue , maxValue) استفاده کنیم. این متد، محدوده مورد نظر را مشخص می‌کند پس کافی است دستور randomNumber. (۱,۵۱) Next را برای رسیدن به مطلوب بنویسیم.

بنابراین دستورهای ما به این صورت نوشته می‌شوند :

```
leftNumber = randomNumber.Next(1, 51);
rightNumber = randomNumber.Next(1, 51);
```

ب) قرار دادن اعداد تصادفی تولید شده در کنترل‌های برچسب :

```
leftLabel.Text = leftNumber.ToString ( );
rightLabel.Text = rightNumber.ToString ( );
```

سؤال: چرا از متد ToString استفاده شده است؟

ج) مقداردهی زمان باقی‌مانده و نمایش پیام :

```
messageLabel.Visible = true;
timeLabel.Text = "5";
timeLeft = 5;
timeLabel.Visible = true;
```

د) فعال کردن زمان‌سنج : زمان‌سنج را برای شروع سنجش، فعال کرده و دکمه «شروع» را برای جلوگیری از رخ دادن رویداد کلیک دکمه تا قبل از پایان این مرحله غیرفعال می‌کنیم :

```
timer1.Enabled = true;
start.Enabled = false;
```

هـ) نمایش و مقدار دهی کنترل NumericUpDown1 :

```
numericUpDown1. Value = 0;
numericUpDown1. Visible = true;
```

متد کامل EH دکمه «شروع» چنین خواهد بود :

```
private void start_Click(object sender, EventArgs e)
{
Random randomNumber = new Random();
leftNumber = randomNumber.Next(1, 51);
rightNumber = randomNumber.Next(1, 51);
leftLabel.Text = leftNumber.ToString();
rightLabel.Text = rightNumber.ToString();
messageLabel.Visible = true;
timeLabel.Text = "5";
timeLeft = 5;
timeLabel.Visible = true;
numericUpDown1.Value = 0;
```

```
numericUpDown1.Visible = true;
timer1.Enabled = true;
start.Enabled = false;
}
```

۱۱ــ نوشتن رویداد Tick برای کنترل زمانی: اکنون باید متد EH رویداد زمانی را بنویسیم. متد مربوطه باید عملیات زیر را انجام دهد :

الف) درستی پاسخ وارد شده به وسیلهٔ کاربر را بررسی کند.

ب) اگر پاسخ کاربر صحیح باشد، پیامی نمایش دهد که «پاسخ شما صحیح است» و سپس رویداد زمانی را از کار بیندازد و دکمه «شروع» را برای سنجش بعدی فعال کند.

ج) اگر پاسخ کاربر نادرست باشد، آنگاه زمان باقیمانده را بررسی کند اگر زمان به پایان رسیده بود، آن‌گاه پیامی‌نمایش دهد که «پاسخ شما اشتباه است» و سپس رویداد زمانی را از کار بیاندازد و دکمه «شروع» را برای سنجش بعدی فعال کند.

د) اگر پاسخ کاربر نادرست باشد، ولی مدت پاسخ‌گویی به سؤال تمام نشده است، آن‌گاه یک ثانیه از زمان باقیمانده کم کند و مدت باقیمانده را روی صفحه نمایش دهد.

```
private void timer1_Tick(object sender, EventArgs e)
{
    if (leftNumber + rightNumber == numericUpDown1.Value)
    {
        timer1.Enabled = false;
        timeLeft = 5;
        MessageBox.Show
                ("سنجش مهارت عمل جمع", " آفرین. شما به این سؤال درست پاسخ دادید");
        start.Enabled = true;
    }
    if (timeLeft > 0)
    {
        timeLeft - -  ;
```

```
            timeLabel.Text = timeLeft.ToString();
        }
        else
        {
            timer1.Enabled = false;
            MessageBox.Show
    ("سنجش مهارت عمل جمع", "پاسخ اشتباه! متأسفانه وقت شما برای پاسخ‌گویی به این سؤال تمام شد");
            start.Enabled = true;
        }

}
```

برای انجام عملیات بالا، بر روی زمان‌سنج، دو بار کلیک کرده تا متد EH مربوطه ظاهر شود و سپس در داخل آن، دستورات زیر را تایپ می‌کنیم.

۳ــ ۱ــ کادر محاوره‌ای پیام

برای بررسی پاسخ کاربر، مجموع دو عدد با پاسخ وارد شده به وسیلهٔ کاربر، مقایسه می‌شود، اگر پاسخ کاربر صحیح باشد، پیام درست بودن پاسخ نمایش داده شده، زمان‌سنج غیرفعال می‌شود و دکمه شروع برای عملیات مجدد فعال می‌گردد. درصورت اشتباه بودن پاسخ کاربر از زمان باقی مانده یک ثانیه کم شده و برچسب‌ها متناسب با آن به روز می‌شوند و در صورت پایان یافتن زمان کاربر پیام مناسب نمایش داده شده، تایمرغیرفعال می‌شود و دکمه شروع برای عملیات مجدد فعال می‌گردد.

در این برنامه برای نمایش پیام از کادر محاوره‌ای MessageBox استفاده شده است. این کادر محاوره‌ای، با استفاده از متد ()Show پیامی‌را در یک پنجره جداگانه نمایش می‌دهد و منتظر می‌شود تا کاربر بر روی دکمه‌ای کلیک کند. شکل ۱۱ــ۱ پیام مربوط به پاسخ صحیح کاربر را نشان می‌دهد.

شکل ۱۱ ــ۱ــ کادر محاوره‌ای پیام

به وسیلهٔ پارامترهایی که به متد ()Show ارسال می‌کنیم، می‌توانیم پیام، عنوان پنجره و دکمه‌های مورد نظر در کادر محاوره‌ای را تعیین کنیم.

MessageBox.Show(متن پیام. عنوان پنجره. دکمه‌ها. آیکون. دکمه پیش‌فرض);

متد Show() دارای شکل‌های مختلفی است که شکل ۱۲ــ۱ یک نمونه استفاده از متد Show() با سه پارامتر را نشان می‌دهد.

شکل ۱۲ــ۱

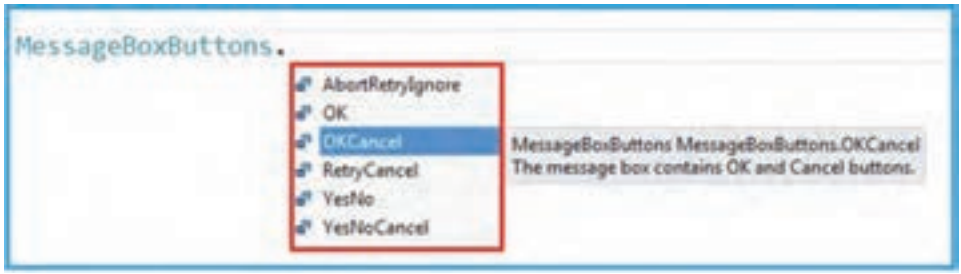
اگر فقط یک پارامتر رشته‌ای به متد Show() ارسال کنید، این رشته در پنجره‌ای بدون عنوان و تنها با یک دکمه "OK" نمایش داده می‌شود .

دکمه‌های پنجره پیام MessageBox را به وسیلهٔ نوع داده شمارشی MessageBoxButtons تعیین می‌کنیم که در فضای نام System.Windows.Forms قرار دارد و اعضای آن در جدول ۱۴ــ۱ آمده است.

جدول ۱۴ــ۱

نام اعضا	شرح
AbortRetryIgnore	کادر پیام شامل دکمه‌های Abort, Retry و Ignore است.
OK	کادر پیام شامل دکمه OK است.
OKCancel	کادر پیام شامل دکمه‌های OK و Cancel است.
RetryCancel	کادر پیام شامل دکمه‌های Retry و Cancel است.
YesNo	کادر پیام شامل دکمه‌های Yes و No است.
YesNoCancel	کادر پیام شامل دکمه‌های Yes, No و Cancel است.

شکل ۱۳ــ۱ اعضای این داده شمارشی را نشان می‌دهد.



شکل ۱۳ــ۱

مثلاً دستور زیر سبب نمایش کادر محاوره‌ای با دو دکمه "yes" و "No" می‌شود.

```
MessageBox.Show ("آیا می خواهید ادامه دهید؟", "سنجش مهارت عمل جمع",
MessageBoxButtons.YesNo);
```

اکنون برنامه را اجرا و خطاهای زمان اجرای آن را بررسی کنید.

توسعه و بهبود برنامه

تمرین : در این قسمت می‌خواهیم برنامه سنجش عمل جمع را توسعه دهیم. برای این منظور عملیات زیر را دنبال می‌کنیم :

الف) توسعه برنامه با نمایش پاسخ صحیح در کنترل NumericUpDown در پایان سنجش :

دستوری به متد EH رویداد زمانی اضافه کنید که در صورت پایان مدت زمان سنجش، پاسخ صحیح را نیز در کنترل عددی افزایشی ــ کاهشی نمایش دهد.

ب) توسعه برنامه با دریافت نام کاربر :

۱ــ در پنجره طراحی فرم، یک کادر متنی (Text Box) و یک برچسب اضافه نمایید تا کاربر نام خود را قبل از شروع سنجش وارد نماید.

۲ــ در دستور MessageBox که پیام مربوط به پاسخ درست و همچنین پیام پاسخ نادرست را نشان می‌دهد، نام کاربر نیز اضافه شود.

ج) توسعه برنامه با اضافه کردن عملیات دیگر به فرم :

علاوه بر سنجش عمل جمع، عملیات دیگر نظیر تفریق ، ضرب و تقسیم نیز اضافه کنید.

راهنمایی : برای عمل تفریق نیز چهار برچسب و یک کنترل عددی افزایشی ــ کاهشی به فرم

اضافه کنید. دو عدد تصادفی نیز تولید کنید. برای اینکه عمل تفریق ساده شود عدد تصادفی دوم همواره باید کمتر یا برابر عدد تصادفی اولی باشد. چرا؟

برای این منظور از دستورات زیر استفاده کنید که این ویژگی را تضمین می‌کند.

```
leftNumber = randomNumber.Next(1, 51);
rightNumber = randomNumber.Next(1, leftNumber+1);
```

د) توسعه و بهبود برنامه با نمایش اطلاعاتی از نتیجه سنجش، شامل تعداد سؤالات، تعداد پاسخ صحیح و تعداد پاسخ غلط :

۱ــ دو فیلد عددی در داخل کلاس فرم برای نگهداری تعداد سؤالات و تعداد پاسخ صحیح با مقدار اولیه صفر تعریف کنید (برای نام متغیرها از totalQuiz و correctQuiz استفاده کنید).

۲ــ در متد EH، دکمه «شروع» مقدار متغیر مربوط به تعداد سؤالات را یک واحد افزایش دهید.

۳ــ در متد EH رویداد زمانی، اگر پس از پایان مدت سنجش، پاسخ کاربر غلط بود، یک واحد به متغیر مربوط به تعداد پاسخ غلط اضافه نمایید.

۴ــ در دستورات MessageBox، باید تغییری دهید که علاوه بر پیام قبلی، تعداد سؤالات و پاسخ‌های درست و نادرست (مقدار متغیرها) را نیز نشان دهد.

خودآزمایی فصل اول

الف) درستی یا نادرستی هر عبارت را تعیین کنید.

۱ــ برای دکمه‌های رادیویی اگر ویژگی RightToleft را Yes انتخاب کنیم، نوشته در سمت راست دایره قرار می‌گیرد.

۲ــ برای انتخاب دکمه رادیویی به‌عنوان پیش‌فرض، ویژگی Enabled آن را True قرار می‌دهیم.

۳ــ کنترل عددی افزایشی ــ کاهشی (NumericUpDown) برای دریافت داده‌های اعشاری هم به‌کار می‌رود.

۴ــ در متد Random.Next(maxValue) مقدار maxValue می‌تواند منفی باشد.

۵ــ از دکمه‌های رادیویی یک groupBox فقط یکی در حالت انتخاب قرار می‌گیرد.

ب) جاهای خالی را با عبارت مناسب پرکنید.

۶ــ از نوع داده شمارشی MassageBoxButtons برای تعیین نوع ..... MassageBox استفاده می‌شود.

۷ــ برای ایجاد دو دسته دکمه رادیویی مستقل از یکدیگر، باید هر دسته از دکمه‌ها را در داخل ..... قرار دهید.

۸ــ برای بستن فرم از متد ..... استفاده می‌شود.

۹ــ متد Random. Next (maxValue) عددی بزرگ‌تر یا مساوی ..... و کوچک‌تر از ..... به ما می‌دهد.

۱۰ــ برای نمایش پیام با استفاده از MessageBox، متد ..... فراخوانی می‌شود.

ج) به سؤالات زیر پاسخ دهید.

۱۱ــ تفاوت کنترل دکمه رادیویی و دکمه انتخاب را بنویسید.

۱۲ــ چرا در فرم‌ها برای تعیین جنسیت از دکمه رادیویی استفاده می‌شود؟

۱۳ــ بررسی کنید اگر در کنترل NumericUpDown ویژگی Increment را با عدد ۲/۵ مقداردهی کنیم، هر بار عدد داخل کنترل چه مقدار افزایش می‌یابد.

برای مقداردهی با اعداد ۲/۱ و ۲/۷ و ۳/۵ هم نتایج را بررسی کنید.

۱۴ــ از نوع داده شمارشی DialogResult به چه منظور استفاده می‌شود؟

۱۵ــ دستوری بنویسید که عدد تصادفی بین ۵ تا ۱۵ تولید کند.

۱۶ــ دستوری بنویسید که عدد تصادفی بین ۱۰ــ تا ۱۰ تولید کند.

۱ــ برنامه ویندوزی بسازید که یک فرم ورود به سیستم (Login) مانند شکل روبرو ایجاد کند. (دکمه‌ها فاقد متد EH هستند.)

۲ــ در تمرین قبل، برای دکمه «ورود» یک متد EH بنویسید. یک نام کاربری و یک رمز عبور دلخواه در برنامه در نظر بگیرید. چنانچه کاربر نام کاربری و کلمه عبور را درست وارد کرد پیام «خوش آمدید» و در غیر این‌صورت پیام «نام کاربری یا کلمه عبور اشتباه است» با استفاده ازکادر محاوره‌ای MessageBox نمایش داده شود. (دکمه «ثبت نام» فاقد متد EH است.)

۳ــ یک فرم برای ثبت‌نام یک کاربر جدید مانند شکل روبرو طراحی کنید. (دکمه‌ها فاقد متد EH هستند.)

۴ــ برای تمرین قبل شرایط زیر را در نظر بگیرید :

الف) نام کاربری حداقل ۵ حرف و حداکثر ۳۰ حرف باشد.

ب) کلمه عبور حداقل ۶ کاراکتر و حساس به حروف کوچک و بزرگ باشد.

ج) کلمه عبور و تکرار باید دقیقاً یکسان باشند.

د) در صورت کلیک بر روی دکمه «ذخیره»، شرایط بالا بررسی شده و در صورت عدم رعایت شرایط مذکور، پیام خطای مناسبی نمایش داده شود و در صورت رعایت تمام موارد، پیام «کاربر جدید با موفقیت به سیستم اضافه شد» با استفاده از کادر محاوره‌ای MessageBox() نمایش داده شود.

هـ) در صورت کلیک بر روی دکمه «پاک کردن»، محتوای کادرهای متنی پاک شد و علامت چشمک زن[1] در داخل کادر متنی نام کاربری قرار گیرد.

۵ــ برنامه ویندوزی بسازید که یک فرم «ارتباط با ما» مانند شکل روبرو ایجاد کند. (دکمه‌ها فاقد متد EH هستند.)

۶ــ درتمرین قبل، برای دکمه‌های فرم، متدهای EH مناسب بنویسید به طوری که :

الف) با کلیک بر روی دکمه «ارسال» درصورت تکمیل بودن فرم، پیام «ارسال موفقیت‌آمیز» نمایش داده شود.

ب) با کلیک بر روی دکمه «انصراف» و در صورت تأیید کاربر، فرم بسته شود.

ج) دکمه «خالی کردن فرم» نیز برای پاک کردن محتوای کادرهای متنی استفاده می‌شود. قبل از پاک کردن، تأیید کاربر توسط یک پیام مناسب گرفته شود.

---

۱ ــ **راهنمایی** : برای قرار گرفتن علامت چشمک زن در کادر متنی مورد نظر، از متد ()Focus کادر متنی استفاده نمایید.

فعالیت

پروژه

در این بخش از فعالیت، برای تمرین و یادگیری بهتر مطالب مطرح شده در کتاب، پروژه‌ای را ایجاد کرده و درهر فصل متناسب با محتوا، برای تکمیل پروژه یک گام به جلو بر می‌داریم. به این ترتیب در پایان کتاب با توجه به دستورات آموزش داده شده، پروژه تکمیل خواهد شد. به این منظور پروژهٔ واژه‌نامه یا فرهنگ لغات معین را انتخاب کرده‌ایم.

در این فصل، فایل پروژه را ایجاد می‌کنیم و آن را برای تکمیل، متناسب با محتوای فصل‌های دیگر، ذخیره می‌کنیم.

مرحله اول

امکان جستجوی معنی واژه، مهم‌ترین امکان هر واژه‌نامه است. در مرحلهٔ اول قصد داریم قسمت جستجوی این واژه‌نامه را با توجه به کنترل‌های ورود اطلاعات وآرایه پیاده‌سازی کنیم.

۱ــ پروژه‌ای از نوع WFA با نام Dictionary ایجاد کنید.

۲ــ با توجه به شکل زیر کنترل‌های لازم را روی فرم قرار داده و ویژگی‌های آنها را تنظیم نمایید.

کنترل‌های فرم اولیه پروژه واژه‌نامه

توجه کنید که کادر معنی، غیرفعال است و امکان نوشتن متن در آن وجود ندارد.

۳ــ برای ایجاد واژه‌نامه باید واژگان و معنی آنها را در بخشی از حافظه ذخیره کنیم، با توجه به انواع داده‌هایی که تاکنون با آن آشنا شده‌ایم می‌توان واژگان و معنی آنها را در آرایه‌ای از نوع String ذخیره کرد. برای سادگی و مطابقت با آنچه تاکنون آموخته‌اید، واژگان و معانی در دو آرایه ۳ عضوی پیاده‌سازی می‌شوند و آرایه در کلاس فرم به صورت زیر تعریف کنید :

```
string[] words = new string [] {"کرامت", "تدبیر", "کنکاش"};
string[] meanings = new []{"۱ـ مشورت، شور، کنکاش هم‌گفته می‌شود (مص م.) به پایان‌کاری)","اندیشیدن. ۲ــ اندیشیدن. ۳ــ مشورت‌کردن. ۴ــ (مص.) پایان‌بینی. ۵ــ شور، (مص ل.) سخاوت،)","جوانمردی، بخشندگی مشورت"};
```

شما می‌توانید از کلمات دیگری به انتخاب خود، استفاده نمایید.

با توجه به تعریف آرایه، مشخص است که عناصر ارایهٔ words واژگان واژه‌نامه و عناصر متناظر آن در آرایهٔ meaning معنی آن واژگان می‌باشد.

۴ــ کد کلیک دکمه جستجو :

```
private void search_Click(object sender, EventArgs e)
{
  bool flag = false;
  for (int i = 0; i < words.Length; i++)
    if (words[i] == searchWord.Text)
    {
      description.Text = meanings[i];
      flag = true;
      break;
    }
  if (!flag)
    MessageBox.Show("پیدا نشد");
}
```

متغیر منطقی flag تعیین می‌کند واژه در واژه‌نامه وجود دارد یا خیر. در صورت وجود واژه، مقدار این متغیر True می‌شود. در پایانِ جستجو، در صورت false بودنِ این متغیر، پیام «پیدا نشد» نمایش داده می‌شود.

برنامه را اجرا کنید و یکی از ۳ واژه موجود مانند «کرامت» را جستجو و نتیجه را مشاهده نمایید. اکنون جستجو را برای واژه «مروت» تکرار کنید.

واژگان و اصطلاحات انگلیسی فصل اول

ردیف	واژه انگلیسی	معنی به فارسی
۱	Cash	
۲	Debit card	
۳	Down	
۴	Entry	
۵	Equal	
۶	Female	
۷	Increment	
۸	Information	
۹	Leave	
۱۰	Male	
۱۱	Maximum	
۱۲	Minimum	
۱۳	Radiobutton	
۱۴	Random	
۱۵	Skill	
۱۶	Sum	
۱۷	Up	
۱۸	Value	

فصل دوم

رویدادهای ماوس و صفحه کلید

در بیشتربرنامه‌های گرافیکی، از ماوس برای رسم اشکال و یا انتخاب ابزارهای گرافیکی موجود در برنامه استفاده می‌شود. برای اینکه بتوانیم برنامه‌های گرافیکی کاربردی بنویسیم، لازم است رویدادهای ماوس را بشناسیم. در کتاب برنامه سازی دو با رویداد کلیک ماوس در برنامه آشنا شده و از آن استفاده کردیم، علاوه بر ماوس، واکنش‌های برنامه نسبت به کلیدهای صفحه کلید قابل کنترل و تنظیم است. در این بخش با رویدادهای دیگری از ماوس و صفحه کلید آشنا می‌شویم.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ رویدادهای مربوط به ماوس و صفحه کلید را توضیح دهد.

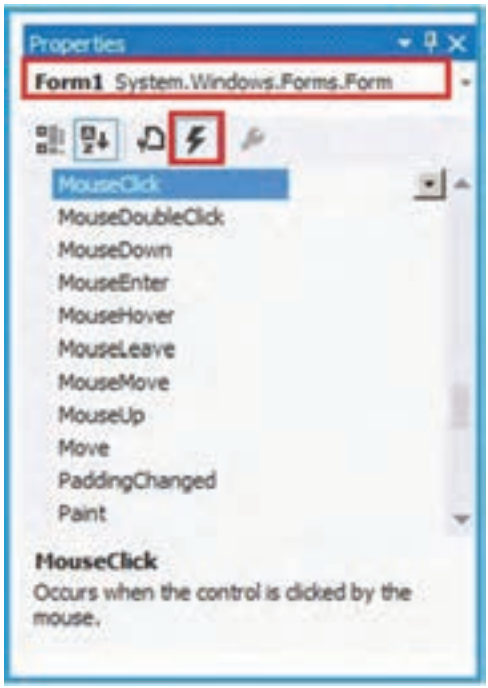
۲ــ از رویدادهای مربوط به ماوس در برنامه‌های کاربردی تعاملی[۱] استفاده نماید.

۳ــ رویدادهای مربوط به صفحه کلید را در برنامه‌های خود به کار بندد.

۱ــ Interactive

۱ــ۲ــ رویدادهای ماوس

اگر یک پروژه WFA ایجاد کنید و در پنجره Properties در قسمت Event مربوط به فرم، لیست رویدادها را مشاهده کنید، در بین آنها چند رویداد مربوط به ماوس است (شکل ۱ــ۲). عملکرد هر رویداد در جدول ۱ــ۲ آورده شده است.



شکل ۱ــ۲ــ رویدادهای ماوس

جدول ۱ــ۲

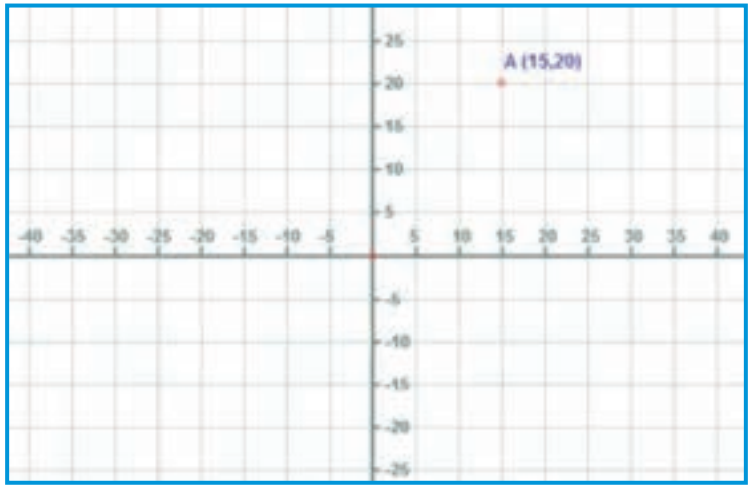
ردیف	نام رویداد	شرح رویداد
۱	MouseClick	کلیک ماوس روی فرم یا کنترل
۲	MouseDoubleClick	دابل کلیک ماوس روی فرم یا کنترل
۳	MouseDown	فشردن دکمه ماوس روی فرم یا کنترل
۴	MouseEnter	ورود از خارج فرم یا کنترل به روی آن
۵	MouseHover	چند لحظه نگه داشتن ماوس روی فرم یا کنترل
۶	MouseLeave	خارج شدن ماوس از روی فرم یا کنترل
۷	MouseMove	حرکت یا تغییر مکان ماوس روی فرم یا کنترل
۸	MouseUp	رها کردن دکمه فشرده شده ماوس

هنگامی که به وسیلهٔ ماوس بر روی فرم کلیک می‌کنید رویداد MouseClick و درصورتی که ماوس را حرکت دهید، رویداد MouseMove رخ می‌دهد. مثال زیر را برای آشنایی با رویداد MouseMove ببینید.

مثال ۱ــ۲ : برنامه‌ای بنویسید که با حرکت ماوس بر روی فرم، موقعیت ماوس با دو عدد طول و عرض نشان داده شود.

شکل ۲ــ۲ــ مختصات ماوس

الگوریتم یا روش انجام کار : از درس ریاضیات به خاطر دارید که هر نقطه در صفحه با دو عدد طول و عرض، نسبت به مبدأ نشان داده می‌شود. به آن دو عدد، **مختصات نقطه**[1] می‌گویند. شکل۳ــ۲ موقعیت نقطهٔ A در صفحه را نشان می‌دهد.



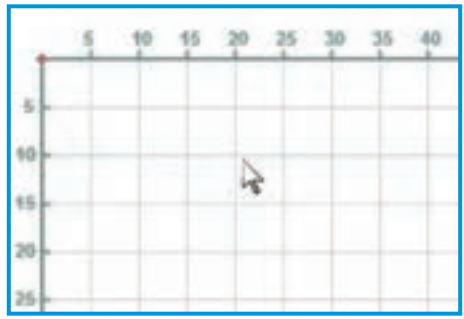
شکل ۳ــ۲ــ موقعیت نقطه در صفحه

۱ــ Coordinates of a point

نکته

در VS نقطه مبدأ برای تعیین مختصات یک شیء یا ماوس بر اساس گوشهٔ بالا سمت چپ در نظر گرفته می‌شود (شکل ۴ــ۲).

موقعیت ماوس در روی فرم با دو عدد سنجیده می‌شود. عدد اول که با حرف X نشان می‌دهیم طول است و عبارت است از فاصله ماوس تا سمت چپ فرم وعدد دوم، فاصله ماوس تا بالای فرم را نشان می‌دهد که به آن عرض نقطه گفته می‌شود و با حرف Y نشان می‌دهیم. معمولاً این دو عدد برحسب پیکسل[۱] است.



شکل ۴ــ۲ــ مبدأ مختصات در صفحه

سؤال: مقدار X و Y اشاره گر ماوس در شکل ۴ــ۲ چقدر است؟

هنگامی که ماوس را حرکت می‌دهید، دو عدد X و Y به وسیلهٔ رویداد MouseMove در اختیار ما قرار می‌گیرد. بنابراین کافی است آنها را در کنترل‌های برچسب قرار دهیم.

کار در کارگاه ۱: نمایش موقعیت (مختصات) ماوس

برای نوشتن برنامه، مراحل زیر را دنبال می‌کنیم :

۱ــ ایجاد یک پروژه ویندوزی : در برنامه VS یک پروژه جدید از نوع Windows Form Application با نام MouseEventDemo در مسیر مشخصی بسازید.

۲ــ تعیین ویژگی‌های فرم : ویژگی‌های فرم را به صورت جدول۲ــ۲ تنظیم کنید.

۱ــ Pixel

جدول ۲ــ۲ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		Form[1]
Text		موقعیت ماوس
Size	Width	300
	Height	300

۳ــ تعیین ویژگی‌های برچسب‌ها : دو کنترل برچسب بر روی فرم قرار دهید و ویژگی آنها را مطابق جداول زیر تعیین کنید.

جدول ۳ــ۲ــ ویژگی‌های کنترل برچسب اول

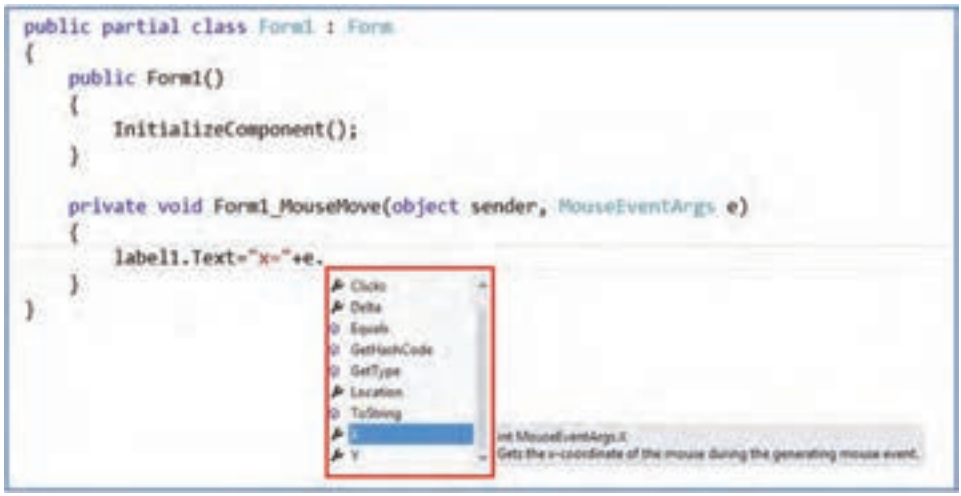
Label		
ویژگی		مقدار
Name		Label1
Text		
AutoSize		True
Location	X	70
	Y	240

جدول ۴ــ۲ــ ویژگی‌های کنترل برچسب دوم

Label		
ویژگی		مقدار
Name		Label2
Text		
AutoSize		True
Location	X	150
	Y	240

۴ــ ایجاد واکنش نسبت به حرکت ماوس : اکنون باید متدی برای واکنش نسبت به رویداد حرکت ماوس بنویسیم. برای این منظور روی فرم، کلیک کنید تا انتخاب شود. سپس در پنجره Properties مربوط به فرم، لیست رویدادها را ظاهر کنید و سراغ رویداد MouseMove بروید. روی آن دو بار کلیک کنید تا پنجره کدنویسی مربوط به متد EH رویداد مورد نظر، ظاهر شود. در داخل متد باید مقدار طول موقعیت ماوس را به وسیلهٔ کنترل برچسب label1 نشان دهیم.

به وسیلهٔ پارامتر e در متد MouseMove، می‌توانیم به جزئیات رویداد ماوس دسترسی داشته باشیم مثلاً می‌توانیم موقعیت ماوس را ببینیم. شکل ۵ــ۲ منوی IntelliSense ویژگی‌های ماوس را نشان می‌دهد. ویژگی X مقدار طول و ویژگی Y مقدار عرض موقعیت ماوس را در اختیار برنامه نویس قرار می‌دهد.



شکل ۵ــ۲

بنابراین دستورات نمایش موقعیت ماوس را مانند زیر می‌نویسیم :

```
private void Form1_MouseMove (object sender, MouseEventArgs e)
{
    label1. Text = "x=" + e.X;
    label2.Text = "y=" + e.Y;
}
```

اکنون می‌توانید برنامه را اجرا کنید و با حرکت ماوس، موقعیت آن را به صورت دو عدد مشاهده کنید.

در کتابخانه .NET کلاسی به نام Cursors تعریف شده که دارای تعدادی ویژگی استاتیک است که هریک از آنها، شکل مختلفی از علامت ماوس را در اختیار قرار می‌دهد. به‌عنوان نمونه، ویژگی Cross شکل ظاهری یا علامتی از ماوس، به‌صورت دو خط متقاطع[۱] (به شکل +) فراهم می‌کند.

۱ــ Cross

از طرفی هر کنترل از جمله فرم، دارای ویژگی به نام Cursor است که مقداردهی آن به یکی از ویژگی‌های کلاس Cursors، سبب می‌شود با قرار گرفتن ماوس در روی آن کنترل یا فرم، علامت ماوس تغییر پیدا کرده و به شکل تعیین شده ظاهر شود. مثلاً اگر می‌خواهید شکل ظاهری ماوس در هنگام حرکت بر روی فرم، به صورت علامت دو خط متقاطع (+) دیده شود، باید ویژگی Cursor فرم را به صورت زیر تغییر دهید :

```
this.Cursor = Cursors.Cross;
```

از کلمه رزرو شده this برای دسترسی به شیء فرم استفاده شده است.

به وسیلهٔ منوی IntelliSense می‌توانید از دیگر ویژگی‌های کلاس Cursors مطلع شوید. اکنون به متد EH مربوط به حرکت ماوس، دستور تغییر شکل ماوس را مانند شکل ۶ــ۲ اضافه می‌کنیم.

شکل ۶ ــ۲ــ منوی IntelliSense کلاس Cursors

```
private void Form1_MouseMove (object sender, MouseEventArgs e)
{
    label1. Text = "x=" + e.X;
    label2.Text = "y=" + e.Y;
    this.Cursor = Cursors. Cross;
}
```

برنامه را مجدداً اجرا کنید و به شکل ظاهری ماوس توجه کنید. در صورتی که بخواهیم شکل ظاهری ماوس، به حالت اولیه (معمولاً علامت فلش) برگردد، از ویژگی Default کلاس Cursors

مطابق دستور زیر استفاده می‌کنیم.

```
this.Cursor = Cursors.Default;
```

سؤال: زمانی که برای فرم، شکل ماوس را تغییر می‌دهید، شکل ماوس برای کدام کنترل‌های روی فرم تغییر می‌کند؟ برای کدام نوع از کنترل‌ها تغییر نمی‌کند؟

کاری که در بالا انجام دادیم را می‌توان در زمان طراحی انجام داد تا خود ویژوال استودیو کد را ایجاد کند. کافی است فرم را انتخاب کنیم و از پنجره properties ویژگی Cursor را برابر Cross قرار دهیم.

اکنون می‌خواهیم زمانی که ماوس روی برچسب‌ها قرار می‌گیرد شکل اشاره گر ماوس به حالت عادی فلش تغییر پیدا کند. برای این کار می‌توانیم از دو روش استفاده کنیم :

۱ــ نوشتن کد به وسیله VS : اگر ویژگی Cursor هر دو برچسب را در پنجره Properties برابر Default قراردهیم کدها به وسیلهٔ VS نوشته می‌شود.

۲ــ نوشتن کد به وسیله برنامه نویس : اگر بخواهیم خودمان کد را بنویسیم به جای استفاده از پنجره Properties دستورات زیر را در رویداد Form_Load می‌نویسیم.

```
label1.Cursor = Cursors.Default;
label2.Cursor = Cursors.Default;
```

رویداد Form_Load زمان بارگذاری فرم در حافظه رخ می‌دهد؛ بنابراین به محض ایجاد فرم شکل اشاره‌گر ماوس برای برچسب‌ها به صورت پیش فرض که همان فلش است، تغییر می‌کند.

نکته

شکل ظاهری ماوس که به وسیلهٔ ویژگی‌های کلاس Cursors فراهم می‌شود، مطابق با شکل‌های تعیین‌شده برای ماوس، در سیستم عامل است. همان‌طورکه می‌دانید به وسیلهٔ تنظیمات ماوس در کنترل‌پنل سیستم‌عامل ویندوز، کاربر می‌تواند شکل‌های استاندارد ماوس را تغییر دهد. به همین دلیل ویژگی Default که در حالت استاندارد علامت فلش است، ممکن است شکل دیگری از ماوس را فراهم

نماید که در این صورت شکل ظاهری ماوس در ویژگی‌های کلاس Cursors نیز مطابق با آن خواهد بود (شکل ۷ــ۲).

شکل ۷ــ۲

برای مطالعه

می‌توان از آرگومان ورودی دوم رویداد MouseMove و رویدادهای دیگر ماوس که ورودی از جنس MouseEventArgs دارند (مانند MouseDown و . . .) استفاده‌های مفید دیگری نیز برد :

e.Location : مکان ماوس در کنترل

e.Button : دکمه‌ای از ماوس که گرفته شده است. فشردن کلیک راست یا چپ و غیره توسط همین خصوصیت قابل شناسایی است.

e.Clicks : تعداد دفعاتی که دکمه ماوس فشرده شده و رها شده است.

کار در کارگاه ۲ــ تونل (راهرو) ماوس

مثال ۲ــ۲ : در این برنامه می‌خواهیم بازی ماز را به شکل ساده شبیه‌سازی کنیم. در فرم ما یک منطقه پیروزی وجود دارد، برای اینکه پیروز شویم باید ماوس را به آن منطقه برسانیم. اما رسیدن به این منطقه باید از مسیر خاص و امن خود باشد وگر نه بازی با شکست مواجه می‌شود.

چگونه این برنامه را طراحی کنیم؟ در این برنامه خواهید دید به راحتی، سازندهٔ یک بازی خواهید شد.

الگوریتم یا روش انجام کار : قبل از هر کاری نقشهٔ راه را پی می‌ریزیم. برای طراحی منطقه پیروزی و راهروها از Label استفاده می‌کنیم. سپس کدهایی را برای انجام عملیات مورد نیاز به برنامه اضافه می‌کنیم. این کدها شامل کد رویدادهایی است که هنگام ورود ماوس به منطقه شکست برای نمایش پیام شکست و رسیدن به منطقه پیروزی برای نمایش پیام پیروزی اجرا می‌شود.

برای نوشتن برنامه، مراحل زیر را دنبال می‌کنیم :

۱ــ ایجاد یک پروژه ویندوزی : وارد برنامه VS شوید و یک پروژه جدید از نوع Windows Form Application با نام MouseCorridor در مسیر مشخصی بسازید.

۲ــ تعیین ویژگی‌های فرم

جدول ۵ــ۲ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		MTunnelForm
Text		راهروی ماوس
Size	Width	400
	Height	400

۳ــ تعیین ویژگی‌های برچسب منطقه پیروزی

جدول ۶ــ۲ــ ویژگی‌های برچسب

Label	
ویژگی	مقدار
Name	goal
Text	پیروزی
AutoSize	false
BackColor	Green
ForeColor	White
TextAlign	MiddleCenter

شکل ۸ــ۲ــ تعیین ویژگی TextAlign

ویژگی TextAlign چینش متن را مشخص می‌کند. برای تغییر ویژگی TextAlign نیز به راحتی می‌توان از کادری که در اختیارمان گذاشته می‌شود استفاده نمود. با استفاده از این کادر محل متن داخل کنترل مشخص می‌شود (شکل ۸ــ۲).

پس از انجام این تنظیم‌ها می توانید اندازه برچسب را مشخص نمایید. این کار را با کشیدن گوشه‌های کنترل Label به وسیلهٔ ماوس یا تنظیم دقیق آن از طریق ویژگی‌های Width (تنظیم عرض) و Height (تنظیم ارتفاع) انجام دهید.

۴ــ تعیین ویژگی‌های برچسب راهروها

جدول ۷ــ۲ ویژگی‌های برچسب

Label	
ویژگی	مقدار
Name	Label1
Text	
AutoSize	false
BackColor	#ff 8000

با انجام چند عملیات Copy و Paste ساده، سایر برچسب‌ها را از روی برچسب اولیه نسخه‌برداری کنید و با جابه‌جا کردنشان، یک راهرو برای عبور ماوس بسازید (مانند شکل۹ــ۲).

شکل ۹ ــ۲ــ فرم بازی تونل ماوس

۵ــ ایجاد واکنش نسبت به حرکت ماوس : در این برنامه اگر ماوس از راهرو امن خود بیرون رود بازی با شکست مواجه می‌شود. رفتن از راهرو به بیرون برابر است با ورود ماوس روی فرم! بنابراین اگر رویدادی باشد که در هنگام ورود ماوس روی فرم، فراخوانی شود می‌توانیم از آن استفاده نماییم. خوشبختانه این رویداد برای کنترل‌های مختلف از جمله فرم وجود دارد. این رویداد MouseEnter نام دارد.

برای استفاده از رویداد MouseEnter در محیط طراحی می‌توانیم به این روش عمل کنیم : ابتدا روی فرم کلیک می‌کنیم تا انتخاب شود. در پنجره Properties دکمه Events را انتخاب می‌کنیم. سپس در گروه Mouse رویداد MouseEnter را پیدا می‌کنیم. با فشردن Enter (یا دو بار کلیک کردن) رویداد مورد نظر ایجاد می‌شود و ما را به محیط کد، انتقال می‌دهد.

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

}
```

در این رویداد، کدی می‌نویسیم تا پیام مناسب برای شکست بازی نمایش داده شود. برای نمایش پیام از متد Show در کلاس MessageBox به صورت زیر استفاده کنید.

```
private void MTunnelForm_MouseEnter(object sender, EventArgs e)
{
    MessageBox.Show("شما بازی را باختید");
}
```

شکل ۱۰ــ۲ــ کادر پیام برنامه

اکنون نوبت آن رسیده است که عملیات اعلام پیروزی را برنامه‌ریزی کنیم. سؤال این است چه زمانی بازیکن پیروز می‌شود؟ زمانی که ماوس وارد منطقه هدف و روی برچسب goal (برچسب پیروزی) وارد می‌شود. بنابراین کلید انجام کار، استفاده از رویداد MouseEnter این کنترل خواهد بود.

این بار نیز می‌توانیم از همان روش قبل برای تعیین رویداد استفاده کنیم. اما روش دوم این است که ابتدا روی برچسب goal کلیک کنیم تا انتخاب شود. در پنجره Properties دکمه Eventes را انتخاب می‌کنیم، سپس در گروه Mouse رویداد MouseEnter را پیدا کرده و نام Victory را در آن می نویسیم. به این ترتیب نام رویداد را خودمان انتخاب نموده‌ایم. با فشردن Enter، رویداد مورد نظر با نامی که وارد کردیم، ساخته خواهد شد. با نوشتن کد پیام پیروزی، کار ما به پایان می‌رسد.

```
private void victory(object sender, EventArgs e)
{
    MessageBox.Show("تبریک..شما برنده شدید");
}
```

وقت اجرای برنامه است! و آنچه مشاهده می کنید این است که با موفقیت توانسته‌اید این بازی را بسازید. توجه کنید برای شروع بازی باید حتماً از نقطه شروع راهرو در سمت چپ فرم استفاده نمایید.

نکته

این برنامه با راهکارهای دیگر هم قابل طراحی است. به عنوان نمونه، شکست بازی به‌جای رویداد MouseEnter فرم، در رویداد MouseLeave کنترل‌های برچسب راهرو نوشته می‌شود. یا اینکه منطقه مجاز حرکت ماوس را فرم و منطقه شکست آن را راهروها درنظر گرفت.

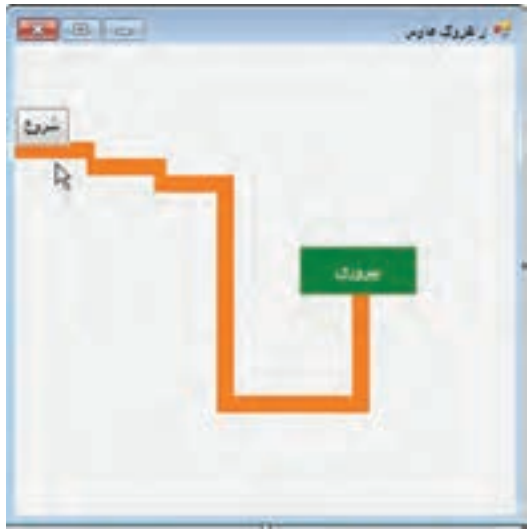
توسعه و بهبود برنامه

با انجام بازی، خواهیم دید هر ورود ماوس به فرم، ایجاد پیام شکست می‌کند. حتی اگر بازی را پیروز شده باشیم به محض آنکه دوباره ماوس را روی فرم بیاوریم پیام شکست نمایش داده می‌شود. این موضوع برای منطقه پیروزی هم صادق است. برای رفع این مشکل چه باید کرد؟ می‌توانیم از یک دکمه استفاده نماییم تا با فشردن آن بازی شروع شود و در صورتی که کاربر پیروز یا بازنده شد، روند بازی متوقف شود و پیام‌ها دیگر نمایش داده نشود. این‌کار مانند فشردن یک کلید برق می باشد. دو حالت وجود دارد : بازی در حال انجام است یا خیر!

هنگامی که از دو حالت نام می بریم می‌دانیم که می‌تواند پای یک متغیر از جنس Boolean در میان باشد! پیشنهاد این است که یک متغیر با نام play و از نوع bool در داخل کلاس فرم تعریف شود تا همه رویدادها و متدهای فرم به آن دسترسی داشته باشند.

```
public partial class MTunnelForm: Form
{
    bool play = false;
    ...
}
```

اگر این متغیر false باشد به معنای آن است که بازی شروع نشده یا تمام شده است و اگر true باشد نشان دهنده این است که بازی در حال انجام است. اکنون دکمه‌ای روی فرم با نام start ایجاد می‌کنیم و متن آن را «شروع» قرار می‌دهیم.



شکل ۱۱ ــ۲ــ فرم بازی با دکمه شروع

با فشردن دکمه، کافی است متغیر play برابر با true شود. چرا؟

```
private void start_Click(object sender, EventArgs e)
{
    play = true;
}
```

در رویداد MouseEnter برای فرم و برچسب goal نمایش پیام بستگی به مقدار متغیر play دارد؛ بنابراین، این رویدادها به صورت زیر تغییر می کند.

```
private void MTunnelForm_MouseEnter (object sender, EventArgs e)
{
    if(play)
    {
        MessageBox.Show("شما بازی را باختید");
        play = false;
    }
    }
```

```
private void victory(object sender, EventArgs e)
{
    if (play)
    {
        MessageBox.Show("تبریک.. شما برنده شدید");
        play = false;
    }

}
```

دلیل بررسی متغیر play برای اجرای دستور پیام چیست؟ این شرط بخاطر آن است که فقط اگر بازی در حالت اجرا باشد (play=true) پیام نمایش داده شود.

سؤال: چرا بعد از دستور نمایش پیام مقدار متغیر play را برابر false قرار می‌دهیم؟

۲ــ۲ــ رویدادهای صفحه کلید

هنگامی که کلیدی از صفحه کلید زده می‌شود، مانند وقتی که ماوس کلیک می‌شود، رویدادهای مختلفی رخ می‌دهد. مثلاً در هنگام فشردن و رها کردن کلیدی از صفحه کلید، سه رویداد مختلف KeyDown، KeyPress و KeyUp به ترتیب رخ می‌دهد که اطلاع می‌دهد چه کلیدی به وسیلهٔ کاربر زده شده است. در روی صفحه کلید، کلیدهایی وجود دارند که با فشردن آنها علامتی یا کاراکتری روی مانیتور ظاهر می شود، به این کلیدها یا کاراکترها، کاراکترهای چاپ شدنی[۱] می گویند. اما بعضی کلیدها نظیر Ctrl، ALT، Shift، Home، End و کلیدهای تابعی مانند F۱ تا F۱۲، علامتی را روی صفحه نشان نمی‌دهد. خوشبختانه به وسیلهٔ رویدادهای ایجاد شده، می توان از فشردن تمام کلیدها مطلع شد.

سه رویداد KeyDown، KeyPress و KeyUp به ترتیب برای اطلاع از پایین آمدن، فشردن کلید، و رها کردن کلیدی ازصفحه کلید استفاده می‌شوند.

هنگامی که رویداد KeyPress رخ می‌دهد، در متد EH مربوطه، می‌توان با استفاده از ویژگی KeyChar مطلع شد که کدام کلید به وسیلهٔ کاربر فشار داده شده است. اما از فشردن اکثر کلیدهای

۱ــ Printable Characters

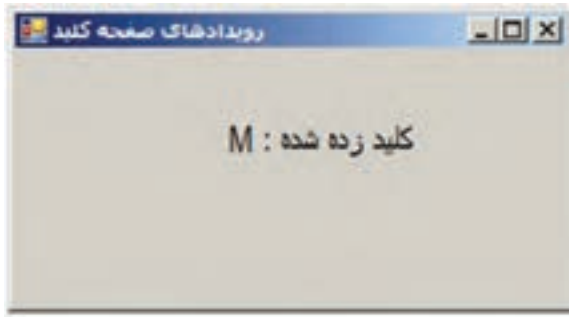
غیرچاپی نمی‌توان مطلع شد و برای این منظور در دو رویداد دیگر KeyDown و KeyUp از فشردن این کلیدها می‌توان خبردار شد.

جدول ۸ــ۲

نام رویداد	شرح رویداد
KeyPress	فشردن یک کلید چاپ شدنی
KeyDown	فشردن یا پایین رفتن کلید
KeyUp	رها کردن کلید فشرده شده

لازم به ذکر است که در زبان‌های غیر انگلیسی، اگر بخواهیم تشخیص دهیم چه حرفی فشار داده شده است باید از KeyPress استفاده کنیم. در دو رویداد دیگر، حرف انگلیسی همان دکمه فشرده شده، قابل شناسایی است.

مثال۳ــ۲ : می‌خواهیم برنامه‌ای بنویسیم که با فشردن کلیدی، کاراکتر مربوطه را با استفاده از یک برچسب نمایش دهد (شکل۱۲ــ۲).



شکل ۱۲ــ۲

الگوریتم یا روش انجام کار : با توجه به توضیحاتی که در مورد رویدادهای صفحه کلید داده شد، کافی است یک برچسب در روی فرم قرار دهیم و برای رویداد فشردن کلید در فرم یک متد EH بنویسیم که در آن کاراکتر کلید فشار داده شده را در درون برچسب نمایش دهیم. بدین منظور مراحل زیر را دنبال می‌کنیم :

۱ــ ایجاد پروژه جدید : وارد برنامه VS شوید و یک پروژه جدید از نوع WFA با نام KeyboardDemo در مسیر مشخصی بسازید.

۲ــ *تعیین ویژگی‌های فرم:* ویژگی‌های فرم را طبق جدول زیر تعیین کنید.

جدول ۹ــ۲ــ ویژگی‌های فرم

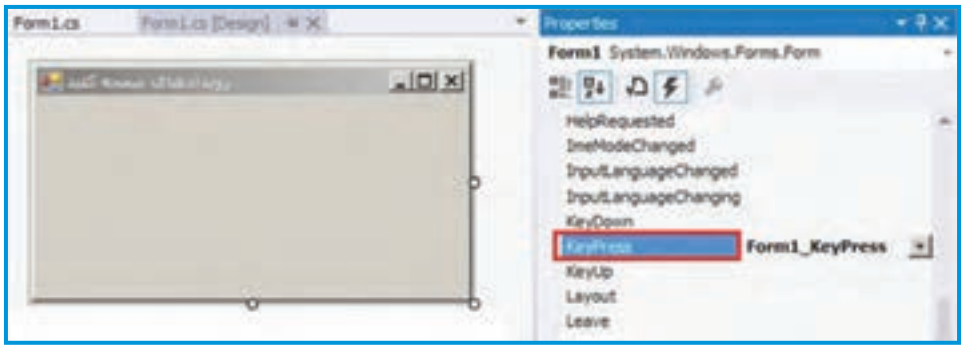
Form		
ویژگی		مقدار
Name		Form1
Text		رویدادهای صفحه کلید
Size	Width	300
	Height	200

۳ــ *تعیین ویژگی‌های برچسب:* یک برچسب به فرم اضافه کنید و ویژگی‌های آن را طبق جدول زیر تعیین کنید.

جدول ۱۰ــ۲ــ ویژگی‌های برچسب

Label		
ویژگی		مقدار
Name		Label1
Text		
AutoSize		True
RightToLeft		Yes
Font		Times New Roman Size: 14
Location	X	100
	Y	30

۴ــ ایجاد واکنش نسبت به صفحه کلید : مطابق شکل ۱۳ــ۲ فرم را انتخاب کرده و در پنجره Properties، رویداد KeyPress را پیدا کرده و دوبار کلیک کنید تا متد EH برای آن ایجاد شود.



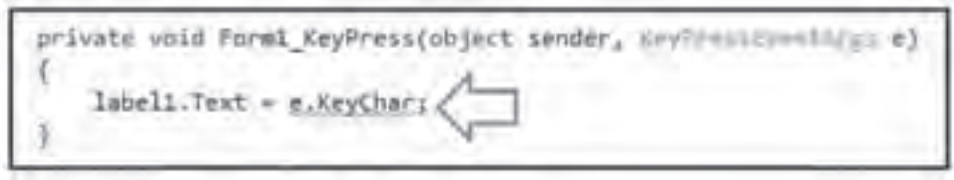
شکل ۱۳ ــ۲ــ ویژگی‌های فرم

متد EH رویداد KeyPress چنین خواهد بود :

```
private void Form1_KeyPress (object sender, KeyPressEventArgs e)
{
    label1.Text = "کلید زده شده :" + e.KeyChar;
}
```

با استفاده از پارامتر e که از نوع KeyPressEventArgs و ویژگی KeyChar، می‌توان کاراکتر متناظر با کلید زده شده را به‌دست آورد، که آن را از طریق برچسب نمایش می‌دهیم. اکنون برنامه را اجرا کنید و کلیدهای مختلفی را بزنید و نمایش آن را روی برچسب مشاهده کنید. کلیدهای غیرچاپی را نیز آزمایش کنید. آیا کاراکتری نشان داده می‌شود؟ آیا برنامه فعلی قادر به تشخیص آن می‌باشد.

سؤال: اگر در کد بالا برای نمایش کاراکتر وارد شده در برچسب، قبل از نمایش ویژگی e.KeyChar، پیام قرار داده نشود، خطا پدید می‌آید (شکل ۱۳ــ۲). چگونه می‌توان این خطا را رفع کرد؟



شکل ۱۴ ــ۲ــ خطای ناشی از مقداردهی نادرست ویژگی برچسب

سؤال: جهت نمایش کد کاراکتر وارد شده، به جای نمایش حرف آن چه تغییری باید در کد برنامه داده شود؟

توسعه و بهبود برنامه

برای اطلاع از فشردن کلیدهای غیر چاپی از رویداد KeyDown استفاده می‌کنیم :

مثال ۴ــ۲ : برنامه مثال قبل را توسعه دهید به طوری که قادر به تشخیص کلیدهای غیر چاپی نیز باشد.

مراحل زیر را برای انجام این کار دنبال می‌کنیم :

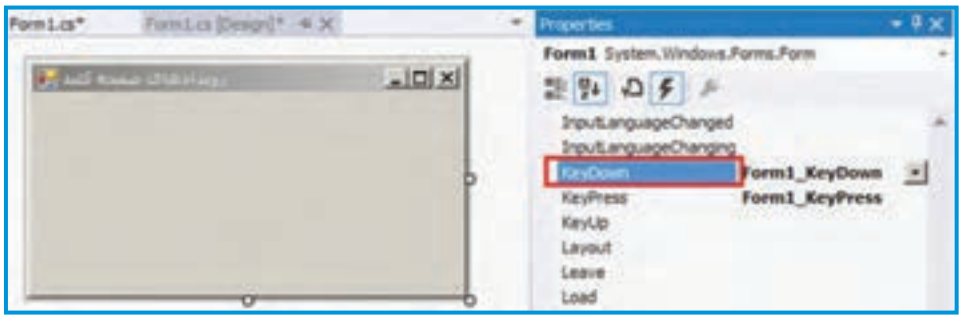
۱ــ پروژه مثال قبل که برای تشخیص کلید فشارداده شده بود را باز کنید.

۲ــ یک برچسب دیگر به فرم اضافه کنید. این برچسب برای اطلاع از اینکه کلید چاپی یا غیر چاپی فشار داده شده به‌کار می‌رود.

جدول ۱۱ــ۲ــ ویژگی‌های برچسب

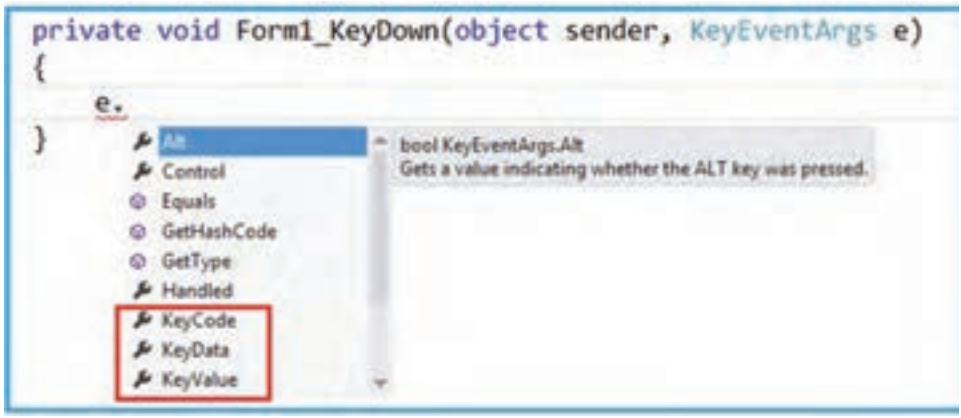
Label		
ویژگی		مقدار
Name		Label2
Text		
AutoSize		True
RightToLeft		Yes
Font		Times New Roman Size: 14
Location	X	**50**
	Y	80

۳ــ مطابق شکل ۱۴ــ۲ فرم را انتخاب کرده و در پنجره Properties، رویداد KeyDown را پیدا کرده و دوبار کلیک کنید تا متد EH برای آن ایجاد شود.



شکل ۱۴ــ۲

۴ــ متد EH رویداد KeyDown دارای پارامتری به نام e از نوع KeyEventArgs است که دارای ویژگی‌های مختلفی است که در شکل ۱۵ــ۲ به وسیلهٔ منوی IntelliSense تعدادی از ویژگی‌های مختلف آن لیست شده است. در جدول ۱۲ــ۲ کاربرد هر یک از ویژگی‌ها را مشاهده می‌کنید :



شکل ۱۵ ــ۲ــ ویژگی‌های پارامتر e

جدول ۱۲ــ۲ــ شرح ویژگی‌های پارامتر e

نام ویژگی	شرح ویژگی
Alt	اگر true باشد یعنی کلید Alt زده شده است.
Control	اگر true باشد یعنی کلید Ctrl زده شده است.
Shift	اگر true باشد یعنی کلید Shift زده شده است.
KeyCode	فقط نام کلید زده شده را اطلاع می‌دهد. اما قادر نیست ترکیب کلیدی زده شده را اطلاع دهد.
KeyData	نام کلید یا ترکیب کلیدی زده شده را اطلاع می‌دهد (مانند Ctrl + P ــ اگر بخواهیم بدانیم کلید همراه کدام کلید کنترلی بوده keydata مناسب است).
KeyValue	عدد صحیحی به عنوان کد کلید زده شده، تولید می‌کند.
SuppressKeyPress	یک متغیر Boolean است. دادن مقدار true به این ویژگی مانع از اجرا و تأثیر کلید روی کنترل می‌شود.

اکنون دستوری برای نمایش کلید یا ترکیب کلیدی زده شده درمتد EH رویداد KeyDown مطابق با شکل می‌نویسیم تا با مقدار خروجی و طریقه استفاده از ویژگی‌های مختلف جدول۱۲ــ۲ آشنا شویم.

```
private void Form1_KeyDown (object sender, KeyEventArgs e)
{
    label2.Text = "KeyData: " + e.KeyData+ "\n" +
                  "KeyCode: " + e.KeyCode+ "\n" +
                  "KeyValue: " + e.KeyValue+ "\n";
}
```

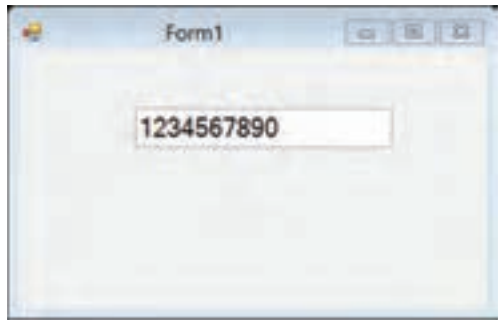
شکل ۱۶ــ۲

اکنون برنامه را اجرا کرده و کلیدهای مختلفی از هر دو گروه چاپی و غیرچاپی را بزنید و عکس‌العمل برنامه را مشاهده کنید. مثلاً اگر کلید shift را نگه داشته و حرف P را بزنید خروجی برنامه چنین خواهد شد :

در این حالت، عدد ۸۰ کد کاراکتر P است. به مقدار خروجی KeyData و KeyCode توجه کنید.

کار در کارگاه۳: ساخت یک کادر متنی عددی

می‌خواهیم کادر متنی ایجاد کنیم که فقط ارقام را بتوان در آن وارد کرد و همچنین کلیدهای Back space و Delete قابل اعمال باشند.



شکل ۱۷ــ۲

الگوریتم و روش انجام کار : برای این کار زمانی که کلیدی زده می‌شود باید بررسی شود اگر کلید زده شده غیر از عدد، Back space یا Delete باشد مانع از اجرای کلید زده شده شود.

۱ــ ***ایجاد یک پروژه جدید:*** در برنامه VS یک پروژه جدید از نوع WF A با نام NumeralTextBox در مسیر مشخصی بسازید.

۲ــ ***تعیین ویژگی‌های کادر متن:*** یک Text Box روی فرم قرار دهید و ویژگی‌های آن را به صورت جدول ۱۳ــ۲ تنظیم کنید.

جدول ۱۳ــ۲ــ ویژگی‌های کادر متن

TextBox		
ویژگی		مقدار
Name		input
Font	Size	14

۳ــ ایجاد واکنش کادر متنی نسبت به صفحه کلید: پس از انجام کارهای مربوط به طراحی واسط کاربری، نوبت کد نویسی است. برای آنکه از کلید زده شده با خبر بشویم و همچنین بتوانیم در صورت لزوم مانع از اجرای کلیدهای خاصی بشویم از رویداد KeyDown کنترل input استفاده می‌کنیم. در این رویداد، بررسی کلید زده شده صورت می‌گیرد. از طریق پنجره properties و قسمت رویدادها، رویداد keyDown را برای کنترل input ایجاد کنید و کدهای زیر را در آن بنویسید:

```
private void input_KeyDown (object sender, KeyEventArgs e)
{
    if (!(char.IsDigit( (Char)e.KeyCode) || e.KeyCode == Keys.Back ||
        e. KeyCode == Keys. Delete) )
        e. Suppresskeypress.true;
}
```

در کد بالا به نکات زیر توجه کنید:

۱ــ متد **IsDigit** برای ساختار char مشخص می‌کند کراکتر ورودی عدد است یا خیر؟ به عنوان نمونه جواب char.IsDigit('1') برابر true خواهد بود.

۲ــ توسط دستور (char)e.KeyCode تبدیل نوع کد کلید به حرف متناظر صورت می پذیرد.

۳ــ نوع شمارشی **Keys** حاوی مقادیر کدها با نام کلید مربوط و مقدار عددی کد آن کلید می‌باشد. به لحاظ شمارشی بودن و نمایش اسامی، کار برای مقایسه کلید زده شده با کلید مورد نظرمان راحت تر می‌شود. در کد بالا وقتی می‌خواهیم ببینیم آیا کلید زده شده Back space است یا خیر، از عبارت منطقی e.KeyCode == Keys.Back استفاده می‌کنیم زیرا Keys.Back نشان دهنده مقدار کلید Back space است. با توجه به همین موضوع، Keys.Delete نمایانگر چه کلیدی است؟

۴ــ با عملگر نقیض "!" که پشت کل عبارت منطقی شرط آمده است مشخص می‌کنیم اگر عبارت منطقی ما درست نباشد (کاراکتر وارد شده،حرف رقم، backspace یا Delete نباشد) مانع از اجرای کلید روی کنترل input شود. دستور **e.SuppressKeyPress=true** این کار را در برنامه انجام می‌دهد.

توسعه و بهبود برنامه

اگر کلیدهای جهت نما را بر روی کنترل کادر متنی امتحان کنیم خواهیم دید که عمل نمی‌کنند چرا؟ با استفاده از نوع شمارشی Keys و مقادیر Keys.Right و Keys.Left می‌توانیم کلیدهای جهت نما راست و چپ را با کلید زده شده مقایسه کنیم. کد رویداد را به صورت زیر اصلاح کنید.

```
private void input_KeyDown (object sender, KeyEventArgs e)
{
    if (!(char.IsDigit( (Char)e.KeyCode) || e.KeyCode == Keys.Back ||
        e. KeyCode = = Keys.Delete ||
        e. KeyCode = = Keys.Right || e.KeyCode == Keys.Left) )
            e.SuppressKeyPress = true;
}
```

سؤال: در نوع داده شمارشی Keys برای دکمه Enter چه مقداری در نظر گرفته شده است؟

خودآزمایی فصل دوم

الف) درستی یا نادرستی عبارات زیر را تعیین کنید :

۱ــ برای ایجاد واکنش نسبت به حرکت ماوس از رویداد MouseMove استفاده می‌شود.

۲ــ با استفاده از ویژگی Cursor در کنترل‌ها می‌توان شکل ماوس را تغییر داد.

۳ــ با استفاده از ویژگی e.KeyChar در رویداد KeyPress می‌توان فهمید کدام کلید فشار داده شده است.

۴ــ با استفاده از رویدادهای KeyDown و KeyUp نمی‌توان از فشردن کلیدهای غیرچاپی مطلع شد.

۵ــ اگر مقدار ویژگی SppressKeyPress یک کنترل False باشد، مانع از اجرا و تأثیرکلید روی کنترل می‌شود.

ب) جاهای خالی را با عبارت مناسب پر کنید :

۶ــ به وسیلهٔ پارامتر e.X و e.Y در متد MouseMove، می‌توانیم به ..... ماوس دسترسی داشته باشیم.

۷ــ برای دسترسی به کد کلید فشاردادهشده در رویداد keydown از ویژگی ..... پارامتر e استفاده می‌شود.

۸ــ اگر بخواهیم ببینیم چه حرفی فشار داده شده، باید از رویداد ..... استفاده کنیم.

ج) جدول زیر را تکمیل کنید.

	عملیات	رویداد پیشنهادی
۹	با ورود ماوس روی کادر متنی، رنگ کادر متنی تغییر کند.	
۱۰	با فشردن دکمه ماوس بر روی کنترل دکمه، متن کنترل تغییر کند.	
۱۱	با نگه‌داشتن ماوس بر روی کنترل قاب (GroupBox) پیامی نمایش داده شود.	
۱۲	با خارج‌شدن ماوس از برچسب، رنگ فرم تغییر کند.	

د) به سؤالات زیر پاسخ دهید.

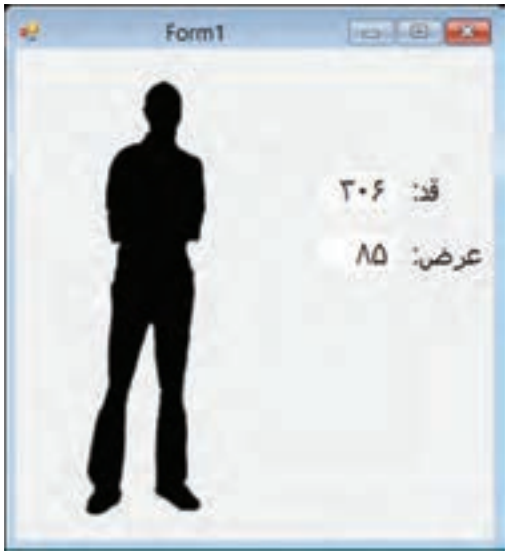
۱۳ــ دستوری بنویسید که در رویداد KeyPress کنترل کادر متنی، کد عددی کاراکتر وارد شده در کنترل برچسب نمایش داده شود.

۱۴ــ برای نمایش کد کلید shift از کدام رویداد صفحه کلید می‌توان استفاده کرد؟

تمرینات برنامه‌نویسی فصل دوم

۱ــ پروژه‌ای بنویسید که کد اسکی کاراکتر وارد شده را روی فرم نمایش دهد. این برنامه را برای کاراکترهای فارسی و کاراکترهای بزرگ و کوچک لاتین و اعداد اجرا نمایید.

۲ــ پروژه‌ای ایجاد کنید که قد و عرض تصویر شخص را با دکمه‌های جهت‌نما تغییر دهد. همچنین قد و عرض را (براساس پیکسل) در Labelهای مربوط درج کند. کارکرد دکمه‌های جهت‌نما به این صورت است :



بالا : افزایش قد (+۱۰)

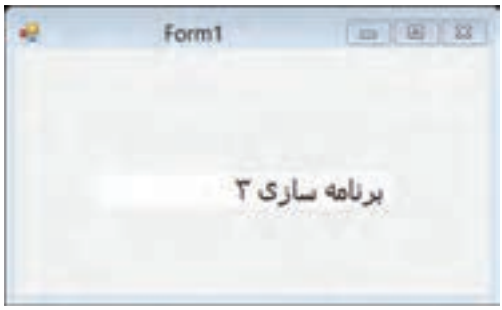
پایین : کاهش قد (۱۰–)

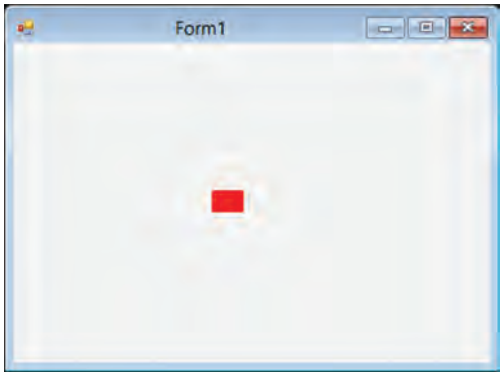
راست : افزایش عرض (۱۰ +)

چپ : کاهش عرض (۱۰ –)

۳ــ فرم ساده‌ای بسازید که با گرفتن یک قسمت داخلی آن (غیر از نوار عنوان)، و کشیدن به یک محل دیگر از صفحه، فرم جابه‌جا شود. درست همان‌طوری که با کشیدن نوار عنوان، این کار انجام می‌شود (از ویژگی‌های left و Top فرم استفاده کنید).

۴ــ فرمی بسازید که دارای یک برچسب باشد که تا حدودی نقش کادر متنی را بازی کند. به این معنا که هر حرفی، به وسیلهٔ صفحه، کلید زده شود به متن برچسب اضافه شود. همچنین اگر Back Space زده شود حرف آخر پاک شود.





۵ــ یک بازی دو نفره طراحی کنید که نفر اول، با دابل کلیک اشاره‌گر ماوس را پنهان کند. آنگاه بازیکن دوم تا ۵ بار فرصت داشته باشد روی محل هدف کلیک کند. درصورت موفقیت، پیام پیروزی ظاهر شود و با کلیک روی هدف یا سپری شدن ۵ بار فرصت کلیک، اشاره گر ماوس ظاهر شود.

برای پنهان شدن اشاره‌گر ماوس از دستور Cursor.Hide() و برای ظاهر شدن آن از دستور Cursor.Show() استفاده کنید.

۶ــ پروژه‌ای بسازید که به‌عنوان یک بازی به صورت زیر استفاده شود. از سمت راست و بالا یک سفینه به سمت چپ حرکت کرده و به تدریج به پایین نزدیک می‌شود. اگر به زمین برخورد کند یک امتیاز کم می‌شود. بازیکن باید به محض دیدن سفینه روی آن کلیک کند تا به این ترتیب سفینه به ارتفاع بالاتری صعود کند تا به تدریج از سمت چپ خارج شود. اگر سفینه از سمت چپ خارج شود یک امتیاز برای بازیکن محسوب می شود. تعداد سفینه‌ها پنج عدد است.

فعّالیت

تکمیل پروژه

در فصل قبل پروژه واژه نامه را ایجاد کردیم و دکمه جستجو را کدنویسی نمودیم. در این فصل رویدادهای ماوس مربوط به این پروژه را تکمیل می‌کنیم.

مرحله دوم:

۱ــ کاربر ممکن است برای جستجو از حروفی استفاده کند که در یک واژه به کار برده نمی‌شود. بنابراین می‌خواهیم اگر از حروف غیر معمول استفاده شود پیام مناسبی ارایه شود.

۲ــ همچنین می‌خواهیم با ورود اشاره‌گر ماوس به قسمت «معنی و توضیح» پس زمینه آن سفید رنگ شود و با خروج اشاره‌گر ماوس، به حالت قبلی برگردد.

ورود اشاره‌گر ماوس

خروج ماوس

۳ــ می‌خواهیم جستجو علاوه بر زدن دکمه، با زدن کلید Enter در کادر واژه نیز انجام شود.

نحوه انجام:

الف) متد EH رویداد KeyPress برای کادر متنی searchWord را این‌گونه معین کنید :

```
private void searchword_KeyPress (object sender, KeyPressEventArgs e)
{
    if (!(char.IsLetter ( e.KeyChar) || e.KeyChar ==' ' || e. KeyChar =='\b' ) )
    {
        MessageBox.Show( "حرف نامعتبری برای واژه وارد شده است");
    }
}
```

شرطی که در کد آمده است با عملگر نقیضی که قبل از پرانتز درونی آمده، بررسی می‌کند اگر هیچ‌کدام از عبارت‌های منطقی زیر صحیح نباشد پیام را نمایش دهد :

- حرف وارد شده جزء حروف الفبایی باشد.
- حرف وارد شده برابر با فاصله باشد.
- حرف وارد شده برابر با Back Space باشد.

نکته

متد IsLetter برای ساختار char بررسی می‌کند که ورودی آن حرف الفبایی است یا خیر.

ب) برای اینکه رنگ پس زمینه با رفتن ماوس روی کادر معنی سفید شود و با خارج شدنش به رنگ قبلی بازگردد، لازم است بتوانیم مقدار رنگ را در متغیری ذخیره کنیم. به نظر شما این متغیر از چه نوعی باید باشد؟ به زبان فارسی نوع «رنگ» و به زبان برنامه نویسی Color است.

بنابراین می‌توان متغیری به صورت زیر تعریف کرد :

```
Color  Bcolor;
```

این متغیر را به صورت سراسری در کلاس فرم تعریف می‌نماییم تا در متدهای مختلف بتوانیم از آن استفاده کنیم و مقدار خود را نیز در طول اجرای برنامه حفظ کند.

ج) متد EH رویداد MouseEnter برای کنترل description را به صورت زیر بنویسید.

```
private void description_MouseEnter(object sender, EventArgs e)
{
    Bcolor = description.BackColor;
    description.BackColor = Color.White;
}
```

همان‌طور که مشخص است در خط اول کد درون متد، رنگ فعلی پس زمینه کادر description به متغیر Bcolor داده می‌شود. با این عمل مقدار رنگ فعلی، برای استفاده بعدی نگهداری می‌شود.

در خط دوم رنگ سفید به مقدار پس زمینه کادر داده می‌شود.

همان‌طور که دیده می‌شود ساختار Color دارای مقادیر مختلفی از رنگ‌ها است. اگر بررسی بیشتری داشته باشید این ساختار، متدی مانند FromArgb برای مقداردهی دقیق رنگ دارد که توسط مقدار چهار مؤلفه یعنی سه رنگ قرمز،سبز، آبی و همچنین مؤلفه میزان شفافیت، رنگ موردنظر را ایجاد می‌کند.

د) متد EH رویداد MouseLeave برای کنترل description را به صورت زیر بنویسید.

```
private void description_MouseLeave(object sender, EventArgs e)
{
    description.BackColor = Bcolor;
}
```

توسط کد این متد، رنگ ابتدایی که در متغیر Bcolor نگهداری شده است را به رنگ پس‌زمینه کادر متنی description نسبت می‌دهیم.

ه) برای اینکه عمل جستجو با زدن دکمه Enter انجام شود کد زیر را می‌نویسیم.

```
private void searchword_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode ==Keys.Enter)
        search.PerformClick( );
}
```

نکته

دستور search.PerformClick() باعث اجرای رویداد کلیک در دکمه search می‌شود.

عمل شماره ۳ همچنین با ویژگی AcceptButton فرم قابل انجام است. در این صورت، دکمه search را برای این ویژگی تنظیم می‌کنیم و نیاز به نوشتن کد ندارد.

واژگان و اصطلاحات انگلیسی فصل دوم

ردیف	لغت انگلیسی	معنی لغت به فارسی
۱	Arrow Key	
۲	Character	
۳	Coordinate	
۴	Cross	
۵	Cursor	
۶	Default	
۷	Demo	
۸	Down	
۹	Event	
۱۰	Height	
۱۱	Hover	
۱۲	Interactive	
۱۳	Leave	
۱۴	Location	
۱۵	Pixel	
۱۶	Play	
۱۷	Point	
۱۸	Press	
۱۹	printable	
۲۰	Tunnel	
۲۱	Up	
۲۲	Victory	
۲۳	Width	

فصل سوم

منو

یکی از وظایف طراحان نرم افزار این است که سیستم‌ها را طوری طراحی کنند که بهره‌برداری از آنها آسان باشد. داشتن منوهای مناسب و سپردن حداقل کار به کاربر، از جمله مسائلی است که به این امر کمک می‌کند. منوها جزء تفکیک ناپذیر هر برنامهٔ خوب محسوب می‌شوند و راهی راحت و پرکاربرد برای دسترسی کاربر به تمام قسمت‌های برنامه را فراهم می‌کنند. برای مثال برنامهٔ ویژوال استودیو با استفاده از منوها، به برنامه نویس این امکان را می‌دهد که به راحتی به ابزارهای این محیط دسترسی داشته باشد و بتواند از آنها استفاده کند. در این فصل می‌خواهیم روش ایجاد منوی مناسب برای برنامه را یاد بگیریم.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ دلایل استفاده از منو را بیان کند و منوی مناسبی برای برنامه خود تعریف نماید.

۲ــ منوهای نواری و زمینه را در واسط گرافیکی کاربر پسند مورد استفاده قرار دهد.

۳ــ تنظیمات ویژه‌ای روی منو و گزینه‌های آن انجام دهد.

۱ــ۳ــ استفاده از منو[1] در واسط گرافیکی کاربر

اغلب برنامه‌های کاربردی مجهز به منو می‌باشند که باعث دسته بندی موضوعی و امکانات یک برنامه می‌شود. با استفاده از منو، کاربر راحت تر می‌تواند به آنچه که مورد نظرش است، برسد. معمولاً گزینه‌های منو به دو روش قابل دسترسی هستند :

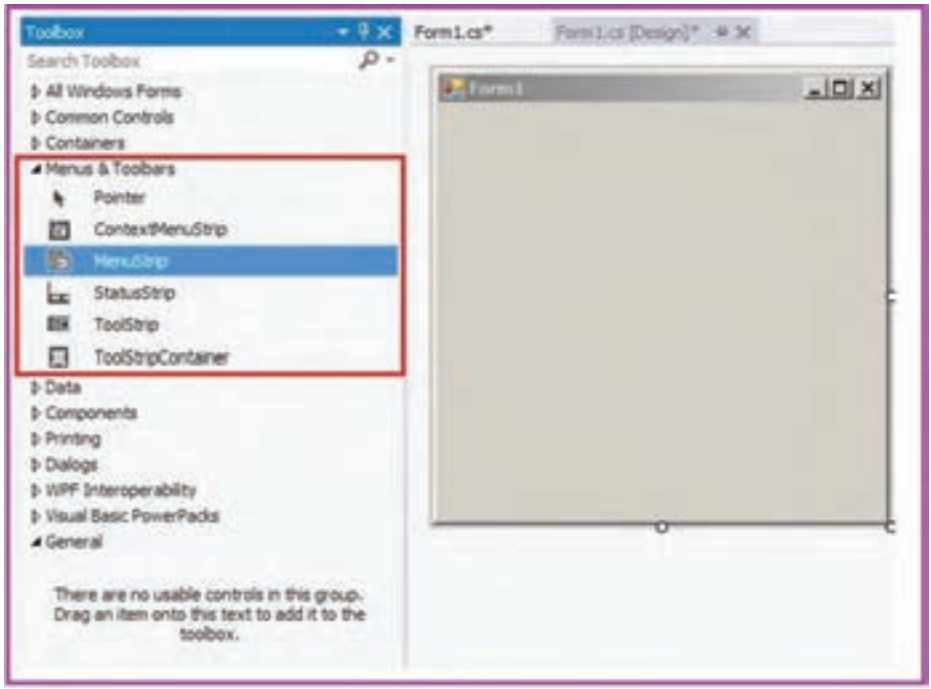
۱ــ وارد کردن یک کلید خاص مانند کلیدهای دسترسی[2] و یا میانبر[3]

۲ــ انتقال مکان‌نما به گزینه موردنظر و فشردن کلید Enter یا کلیک ماوس

اکنون برای اضافه‌کردن منو به برنامه، مراحل زیر را دنبال کنید :

۱ــ برای ایجاد منو، یک پروژه جدید از نوع ویندوز بسازید.

۲ــ در محیط VS، در حالی که پنجره طراحی فرم دیده می‌شود سراغ پنجره جعبه ابزار رفته و در جعبه ابزار فرعی Menu & Tool Bars ابزار MenuStrip را پیدا کنید و آن را به فرم اضافه نمایید (شکل ۱ــ۳).



شکل ۱ــ۳ــ نحوهٔ دسترسی به ابزار منو

۱ــ Menu

۲ــ Hot Key

۳ــ Shortcut Key

ازنظر ظاهری با اضافه‌کردن ابزار منو (MenuStrip) به فرم، دو اتفاق می‌افتد :

۱ــ یک شیء به نام menuStrip1 در زیر فرم قرار می‌گیرد.

۲ــ در زیر خط عنوان پنجره فرم، نوار منویی قرار می‌گیرد که در داخل آن می‌توانید گزینه‌های منوهای افقی، عمودی و زیر منوهای مورد نظر خود را تایپ کنید (شکل ۲ــ۳).

شکل ۲ــ۳ــ نحوهٔ کار با شیء منو

۳ــ در اولین عنوان، عبارت Type Here ظاهر می‌شود، برای تغییر نام، روی این عنوان کلیک کرده، متن جدید را تایپ و کلید Enter را فشار دهید. با استفاده از کلیدهای مکان نما یا ماوس می‌توان بین گزینه‌های منو حرکت کرده، گزینه مورد نظر را انتخاب کرد و عنوان آن را تغییر داد.

۴ــ زمانی که بر روی شیء menuStrip1 کلیک می‌کنیم، پنجره properties، ویژگی‌های آن منو را نمایش می‌دهد و مانند هر کنترل دیگری می‌توانید ویژگی‌های آن را تنظیم کنید (جدول۱ــ۳).

جدول ۱ــ۳ــ ویژگی‌های متداول شیء منو

ویژگی	شرح
Cheked	تیک دار کردن گزینه منو
Enabled	فعال یا غیرفعال کردن گزینه منو
Image	قرار دادن تصویر برای گزینه منو
Name	نام مربوط به منو یا گزینه‌های منو در کد نویسی
RightToLeft	راست به چپ کردن گزینه‌های منو (مانند زبان فارسی)
ShortCutKeys	تعریف کلید میانبر
Text	گزینه‌های منو
Visible	نمایش گزینه‌های منو

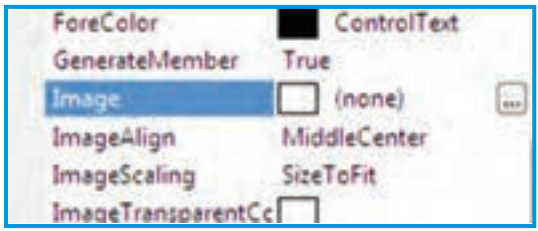
۲ــ۳ــ امکانات منو

برای منوهایی که به وسیله این کنترل ایجاد می‌کنید علاوه بر عنوانی که می‌توانید برای هر منو و گزینه‌های منو نمایش دهید، می‌توانید امکاناتی شامل آیکن (تصویر)، کلید دسترسی، کلید میان‌بر و یا حتی گزینه‌هایی با امکان قابلیت انتخاب را به‌کار گیرید (شکل۳ــ۳).

شکل ۳ــ۳ــ امکانات منو

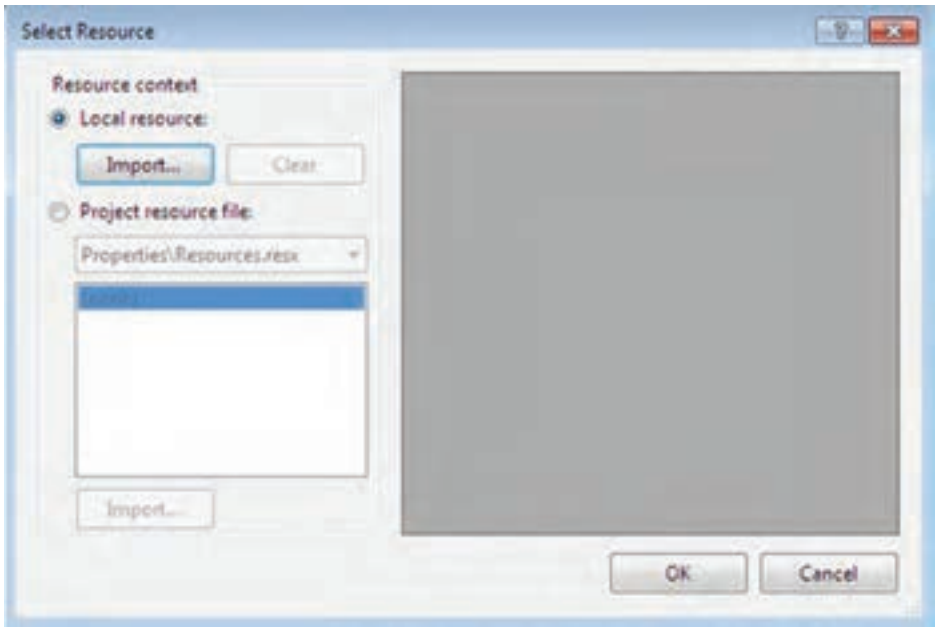
۱ــ۲ــ۳ــ جدا کننده بین گزینه‌ها : جداکننده[۱] به خط افقی گفته می‌شود که با قرار گرفتن در بین گزینه‌های منو، آنها را دسته‌بندی می‌کند. برای اضافه‌کردن جداکننده به گزینه‌های منو، باید در هنگام طراحی منو، در قسمت نام گزینه منو علامت – (منها) قرار داد.

۲ــ۲ــ۳ــ اضافه کردن آیکن به منوها : شما می‌توانید برای هر گزینه منو، یک تصویر کوچک را انتخاب کنید. که در سمت چپ آن نمایش داده شود برای این کار از خاصیت Image استفاده می‌شود. گزینه مورد نظر در منو را انتخاب نمایید. بر روی دکمه کوچک کنار خاصیت Image در پنجره Properties کلیک کنید.



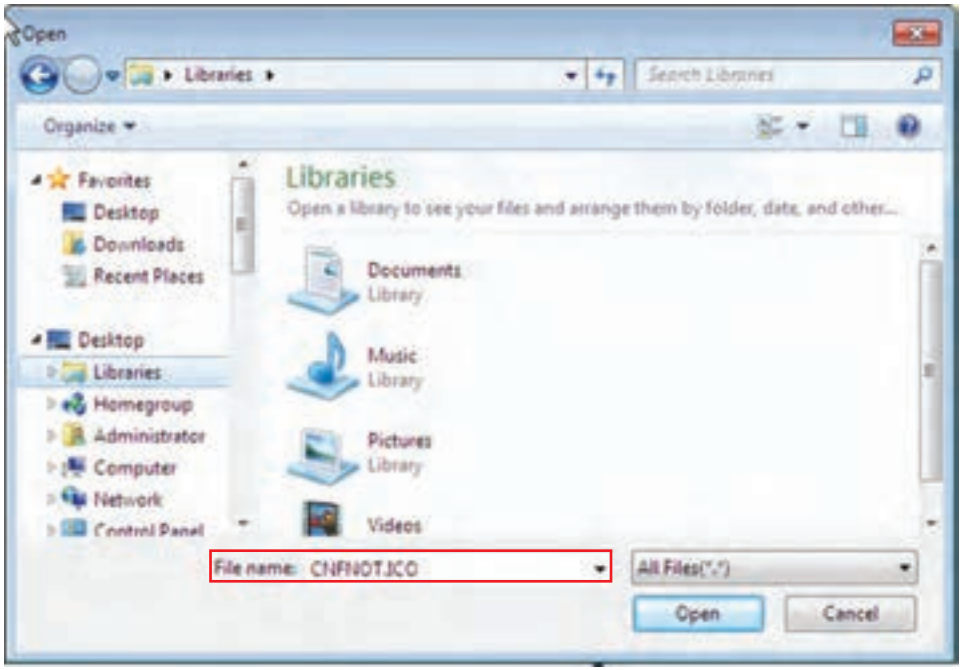
شکل ۴ــ۳ــ تعیین ویژگی Image گزینه منو

منبعی که عکس در آن قرار دارد را انتخاب و سپس مسیر آن‌را مشخص نمایید. اگر اندازه عکس بزرگ باشد، به صورت خودکار بر اساس مقدار خاصیت ImageScaling اندازه آن تغییر می‌کند.



شکل ۵ ــ۳ــ اضافه کردن تصویر به منابع پروژه با استفاده از دکمه Import

۱ــ Separator



شکل ۶ ــ۳ــ انتخاب تصویر از نوع ico. برای آیکن منو

نکته

برای آیکن (تصویر گزینه) هم از فایل‌های آیکن (ico.) و هم از تصاویر دلخواه (با فرمت‌های wmf.,gif.,bmp.,png.,jpg.) می‌توان استفاده کرد.

۳ــ۲ــ۳ــ ایجاد کلید دسترسی : کلیدهای دسترسی، آن دسته از کلیدها هستند که برای فعّال کردن منو و گزینه‌های آن با استفاده از صفحه کلید تعریف می‌شوند. که با فشردن هم‌زمان Alt و یک حرف خاص منجر به بازشدن منو می‌شوند.

برای ایجاد کلید دسترسی، قبل از کاراکتر مورد نظر از علامت & استفاده می‌کنیم.

&Edit ← نمایش ← Edit

در مثال بالا منوی Edit با استفاده از کلیدهای Alt+E قابل دسترسی است.

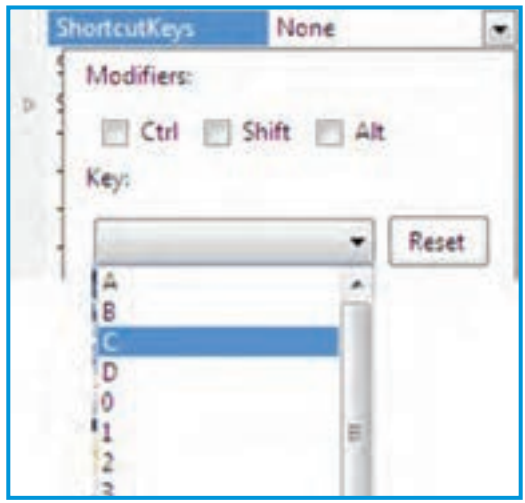
نکته

در زمان اجرای برنامه با فشردن کلید Alt، می‌توان کلیدهای دسترسی منوی تعریف شده را مشاهده کرد.

سؤال: آیا می‌توان از حروف فارسی به عنوان کلید دسترسی استفاده نمود؟

۴ــ۲ــ۳ــ ایجاد کلید میان‌بر : با استفاده از ویژگی ShortCutKeys، می‌توان کلید میان‌بر به منو اضافه کرد که باعث می‌شوند تا کاربران بتوانند عملکرد گزینه‌های منو را با سرعت بیشتری انجام دهند.

در پنجره properties در قسمت ShortcutKeys می‌توان کلید ترکیبی مورد نظر را انتخاب نمود. کلیدهای ترکیبی از ترکیب کلیدهای Ctrl یا Shift یا Alt به همراه حروف الفبا یا ارقام درست می‌شوند (شکل ۷ــ۳).



شکل ۷ــ۳ــ نحوه ایجاد کلید میان‌بر

سؤال: در استفاده از کلیدهای میان‌بر آیا بین حروف کوچک و بزرگ تفاوت وجود دارد؟

نکته

در انتخاب کلید ترکیبی برای کلید میان‌بر دقت کنید تا با کلیدهای میان‌بر ویندوز تداخل نداشته باشد.

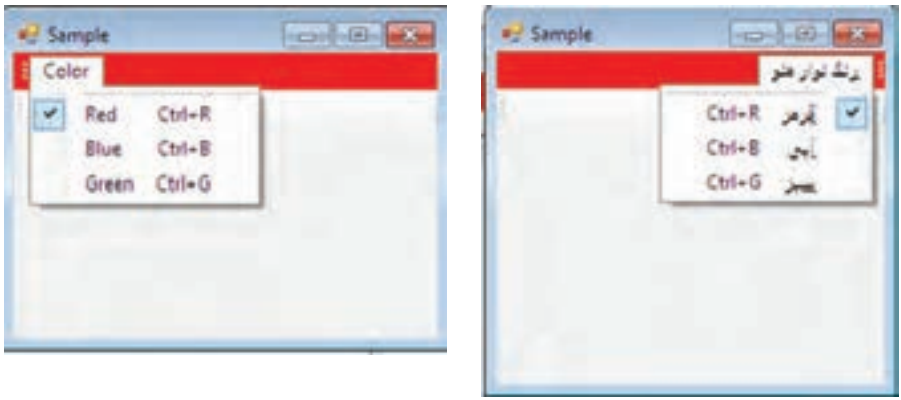
سؤال: در صورت انتخاب کلید میان‌بر مشترک با ویندوز، کدام دستور اجرا می‌شود ؟

کار در کارگاه ۱

مثال ۱ــ۳ : فرم‌طراحی کنید که با استفاده از گزینه‌های منو، رنگ نوار منو تغییر کند.

۱ــ درج کنترل منو : یک شیء از کنترل MenuStrip برای برنامه ایجاد می‌کنیم.

۲ــ تعیین نام و گزینه‌های منو : یک منو با نام Color ایجاد کرده و سپس برای آن چند گزینه رنگ (مثلاً Red , Blue , Green) در نظر می‌گیریم (شکل ۸ــ۳).



شکل ۸ــ۳ــ اجرای منوی طراحی شده برای مثال ۱ــ۱

۳ــ تعیین ویژگی منو : برای گزینه‌ها، کلید دسترسی و کلید میان‌بر تعریف می‌کنیم. برای علامت‌دار شدن گزینه مورد نظر از ویژگی Checked کمک می‌گیریم. می‌توان منوی فارسی ایجاد کرد. برای جهت آن از راست به چپ مقدار ویژگی Right TO Left را Yes در نظر می‌گیریم.

جدول ۲ــ۳ــ ویژگی‌های منو

ویژگی	مقدار		
name	RedToolstripMenuItem	blueToolstripMenuItem	greenToolstripMenuItem
Checked	True	false	false
ShortcutKey	Ctrl+R	Ctrl+B	Ctrl+G
Text	Red	Blue	Green

۳ــ۳ــ طراحی منوی استاندارد

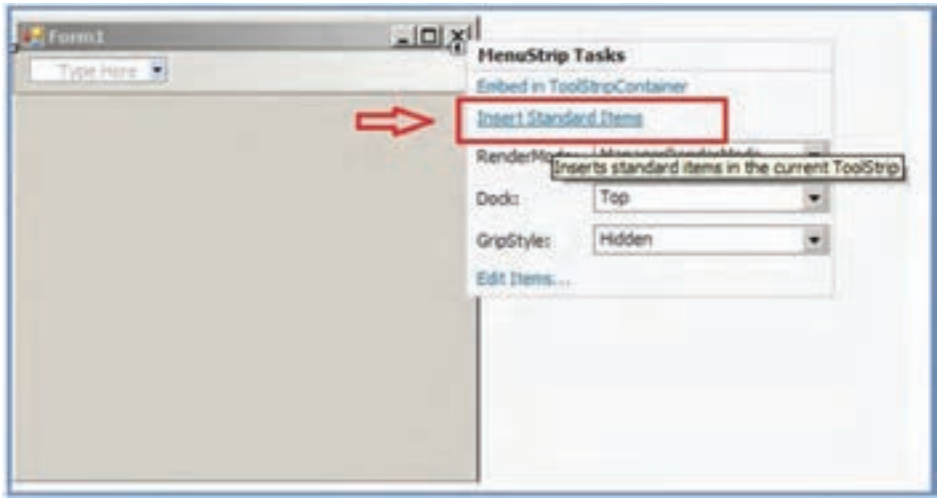
علاوه براینکه می‌توانید منوی دلخواه خود را بسازید، اگر مایل باشید می‌توانید منوی‌های استانداردی که معمولاً در برنامه‌های کاربردی ویندوز دیده می شود، به برنامه خود اضافه کنید.

مثال ۲ــ۳ : چگونه منوهای استاندارد به فرم خود اضافه کنیم؟

الگوریتم یا روش انجام کار :

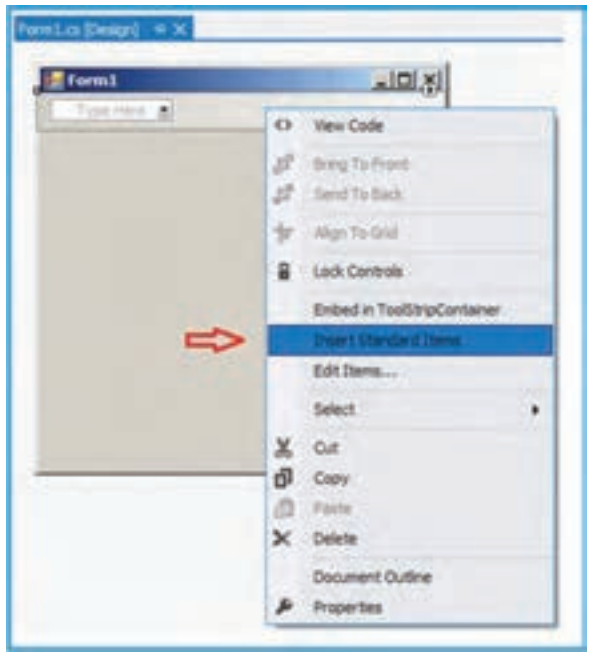
درج کنترل منو : ابتدا کنترل MenuStrip را به فرم اضافه می کنیم.

فراخوانی منوی استاندارد : کنترل منو را انتخاب و سپس روی مثلث مشکی در کادر آن کلیک کرده، لیستی ظاهر می‌شود. گزینه Insert Standard Items را انتخاب می کنیم (شکل ۹ــ۳).



شکل ۹ــ۳ــ نحوه انتخاب منوی استاندارد

همچنین می‌توانید به روش دیگر، بر روی نوار منوی MenuStrip کلیک راست کرده و از لیستی که ظاهر می‌شود، گزینه Insert Standard Items را انتخاب کنید (شکل ۱۰ــ۳).

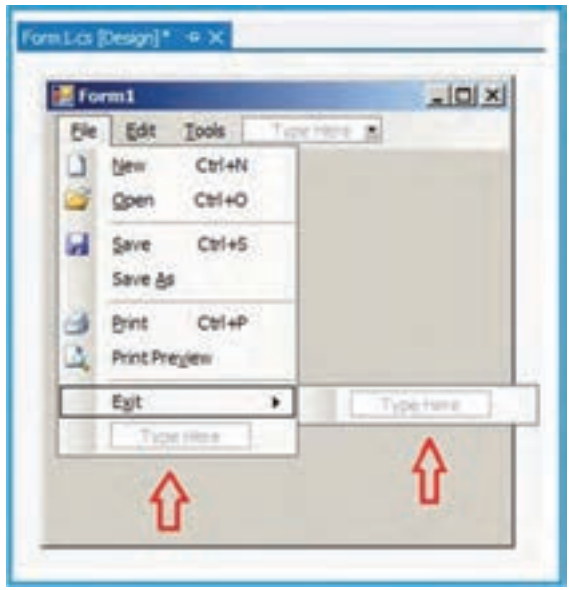


شکل ۱۰ــ۳ــ نحوه انتخاب منوی استاندارد با استفاده از کلیک راست روی نوار منو

در هریک از دو روش، منوی استاندارد به فرم مانند شکل ۱۱ــ۳ اضافه می‌شود.

شکل ۱۱ــ۳ــ نمایی از منوی استاندارد

پس از انجام عمل فوق، در بالای فرم، منوهای File، Edit، Tools، Help مشاهده می‌شود که در هر یک از آنها، گزینه‌های فرعی وجود دارند. در صورت نیاز می‌توانید این منو را بر طبق نیاز، تغییر دهید و گزینه‌های جدیدی را اضافه و یا حذف کنید (شکل ۱۲ــ۳).



شکل ۱۲ــ۳ــ گزینه منوی File از منوی استاندارد

با اجرای برنامه، منوهای استاندارد به همراه گزینه‌های فرعی آنها، دیده می‌شود اما با کلیک برروی هر یک از گزینه‌ها، اتفاقی نمی‌افتد. چرا؟

۴ــ۳ــ واکنش نسبت به گزینه‌های منو

برای فعال‌سازی گزینه‌های منو باید کدهای مربوط به آنها را نوشت. برای نوشتن کد مربوط به هر گزینه، روی آن دوبار کلیک کرده تا وارد محیط کدنویسی شده، دستور (دستورات) مورد نظر را بنویسیم. از متد EH برای این کار استفاده می‌کنیم. با توجه به اینکه عملیات ثبت رویداد و برقرارکردن ارتباط بین رویداد کلیک و متد EH به وسیلهٔ VS به طور خودکار انجام می‌شود. بنابراین کار دیگری به جز اجرای برنامه باقی نمی‌ماند.

مثال ۳ــ۳ : گزینه Exit در منوی File را برای خاتمه دادن به برنامه کدنویسی کنید.

الگوریتم یا روش انجام کار : باید یک متد EH برای گزینه Exit بنویسیم. بنابراین در پنجره طراحی فرم، با دوبار کلیک بر روی گزینه Exit، پنجره کد نویسی باز می‌شود در داخل متدی که برای این منظور توسط VS ایجاد شده است، دستور بستن پنجره را می‌نویسیم :

```
public partial class Form1 : Form
{
    public Form1( )
    {
        InitializeComponent ( );
    }
    private void exitToolStripMenuItem_Click(object sender,
    EventArgs e)
    {
        this.Close( );   ⇦ دستور بستن پنجره
    }
}
```

مثال ۴ــ۳ : می‌خواهیم برای مثال ۱ــ۲ که طراحی کردیم کدنویسی کنیم به‌طوری‌که با انتخاب رنگ مورد نظر، رنگ نوار منو تغییر کند و گزینه مورد نظر علامت‌دار شود.

الگوریتم یا روش انجام کار :

۱ــ نوشتن رویداد Click برای گزینه Red : روی گزینه "Red" دو بار کلیک کرده و کد زیر را وارد می‌کنیم :

```
private void redToolStripMenuItem_Click(object sender, EventArgs e)
{
    menuStripl. BackColor = Color. Red;   ⇦ تعیین رنگ نوار منو
    colorStripMenuItem.BackColor = Color.Red;   ⇦ تعیین رنگ گزینه منو
```

```
        redToolStripMenuItem.Checked = true;
        blueToolStripMenuItem.Checked = false;
        greenToolStripMenuItem.Checked = false;
    }
```

۲ــ نوشتن رویداد Click برای بقیه گزینه‌ها : برای گزینه‌های دیگر نیز مطابق با کد بالا می‌توان کدهای مربوط به رنگ "Green" و "Blue" را نیز نوشت.

مثال ۵ ــ۳ : پروژه‌ای ایجاد کنید که دارای سه منوی File و Edit و View باشد. دو کادر متنی روی فرم در نظر بگیرید. منوی File دارای گزینه New برای خالی کردن کادرهای متنی و گزینه Exit برای خروج از برنامه است. منوی Edit دارای دو گزینه Copy و Paste و منوی View دارای گزینه BackColor و گزینه ForeColor برای تعیین رنگ زمینه و نوشته‌های کادر متنی با استفاده از کادر محاوره‌ای رنگ است.

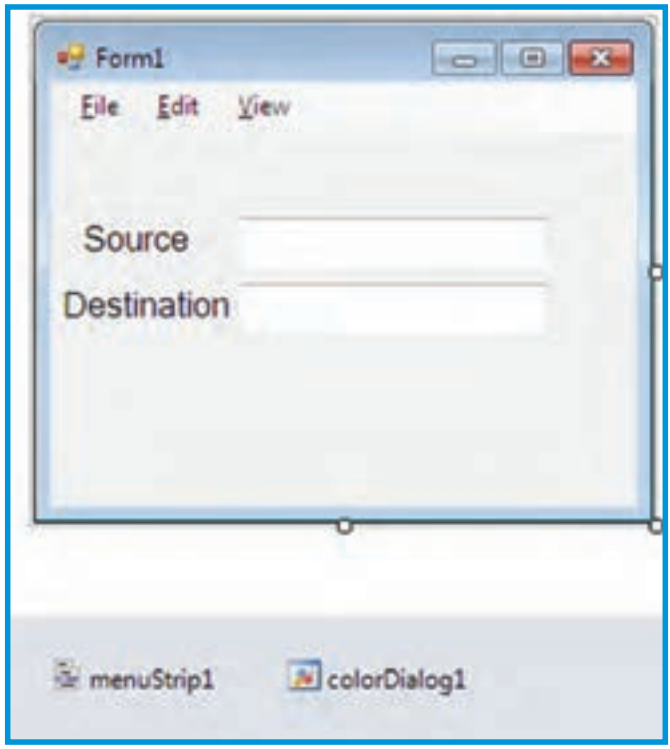
الگوریتم یا روش انجام کار :

۱ــ درج کنترل منو : با استفاده از ابزار Menu & Tool Bars یک شیء MenuStrip برای فرم ایجاد می‌کنیم و منوی استاندارد را برای آن فرا می‌خوانیم.

۲ــ ایجاد گزینه‌های مورد نظر برای منو : در منوی File ، گزینه‌های New و Exit را نگه داشته و بقیه گزینه‌ها را حذف می‌کنیم. در منوی Edit، به غیر از گزینه‌های Copy و Paste بقیه گزینه‌ها را حذف می‌نماییم. منوی Tools را به View تغییر نام داده و گزینه‌های قبلی را حذف کرده و دو گزینه BackColor و ForeColor را اضافه می‌نماییم. منوی Help را نیز حذف می‌کنیم.

۳ــ درج کنترل‌های کادر متنی : دو کادر متنی روی فرم قرار می‌دهیم.

۴ــ درج کنترل ColorDialog : از قسمت ابزار Dialogs کادر محاوره‌ای ColorDialog را برای فرم در نظر می‌گیریم (شکل ۱۳ــ۳ و شکل ۱۴ــ۳).



شکل ۱۴ــ۳ــ فرم مثال ۵ــ۳

شکل ۱۳ــ۳ــ جعبه ابزار Dialogs

۵ــ نوشتن رویداد Click برای گزینه New : روی گزینه New دو بار کلیک کرده و کدهای زیر را می‌نویسیم :

```
textBox1.Text = " ";
textBox2.Text = " ";
```

۶ــ نوشتن رویداد Click برای گزینه Exit : روی گزینه Exit دو بار کلیک کرده و کد زیر را می‌نویسیم :

```
this.Close();
```

۷ــ متغیر سراسری s1 را در داخل کلاس Form1 تعریف کنید :

```
string s1;
```

۸ــ نوشتن رویداد Click برای گزینه Copy : روی گزینه Copy دو بار کلیک کرده و کد زیر را می‌نویسیم :

```
S1 = textBox1.SelectedText;
```

۹ــ نوشتن رویداد Click برای گزینه Paste : روی گزینه Paste دو بار کلیک کرده و کد زیر را می‌نویسیم :

```
textBox2.Text = s1;
```

سؤال: کدهای گزینه Copy را طوری تغییر دهید که اگر متنی انتخاب نشده پیغام خطا بدهد.

۱۰ــ نوشتن رویداد Click برای گزینه BackColor : روی گزینه BackColor دو بار کلیک کرده و کدهای زیر را می‌نویسیم :

```
colorDialog1.ShowDialog( );
textBox1.BackColor = colorDialog1.Color;
textBox2.BackColor = colorDialog1.Color;
```

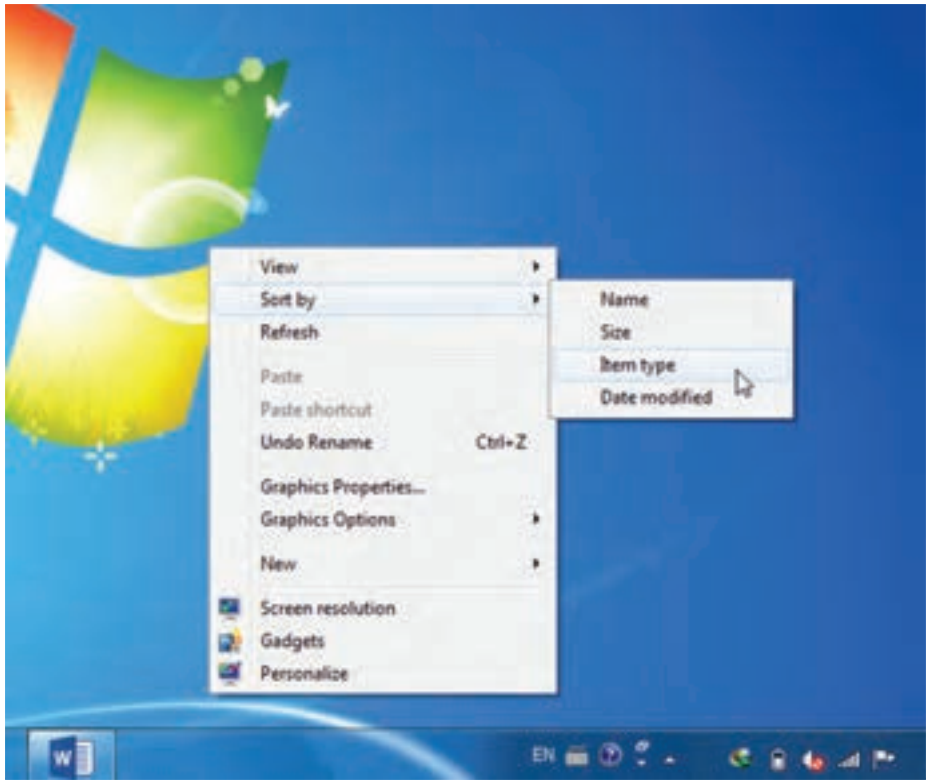
۱۱ــ نوشتن رویداد Click برای گزینه ForeColor : روی گزینه ForeColor دو بار کلیک کرده و کدهای زیر را می‌نویسیم :

```
colorDialog1.ShowDialog();
textBox1.BackColor = colorDialog1.Color;
textBox2.ForeColor = colorDialog1.Color;
```

سؤال: چرا بهتر است برای تغییر گزینه منو به جای تغییر نام، آن گزینه را حذف کرده و گزینه جدید تعریف شود؟

۵ــ۳ــ منوی زمینه[۱]

شکل ۱۵ــ۳ را بارها روی میز کار ویندوز دیده‌اید. این‌گونه منوها که با کلیک راست بر روی زمینه پدید می‌آیند، منوی زمینه یا میان‌بر نام دارند. این منوها در هر موقعیت مکانی از صفحه، ممکن است دارای گزینه‌های متفاوتی باشند.



شکل ۱۵ــ۳ــ منوی زمینه با کلیک راست روی دسک تاپ ویندوز

کنترل منوی زمینه در گروه Menu & Tool Bars از جعبه ابزار (ToolBox) قرار دارد. نام این کنترل ContextMenuStrip است. نحوه استفاده از این کنترل را در مثال صفحهٔ بعد ببینید.

۱ــ Context Menu

مثال ۶ــ۳ : می‌خواهیم یک ویرایشگر متنی داشته باشیم که فرمان تغییر نوع قلم، رنگ متن و رنگ پس زمینه آن از طریق منوی کلیک راست قابل دسترسی باشد (شکل ۱۶ــ۳).

شکل ۱۶ــ۳ــ ویرایشگر متنی با منوی زمینه

الگوریتم یا روش انجام کار :

۱ــ ایجاد پروژه : یک پروژه جدید از نوع WFA با قراردادن یک کادر متنی در فرم ایجاد نمایید.

۲ــ تنظیم ویژگی‌های کادر متنی

جدول ۳ــ۳ــ ویژگی‌های کادر متنی

Text Box	
ویژگی	مقدار
Name	editor
Dock	fill
Multiline	True

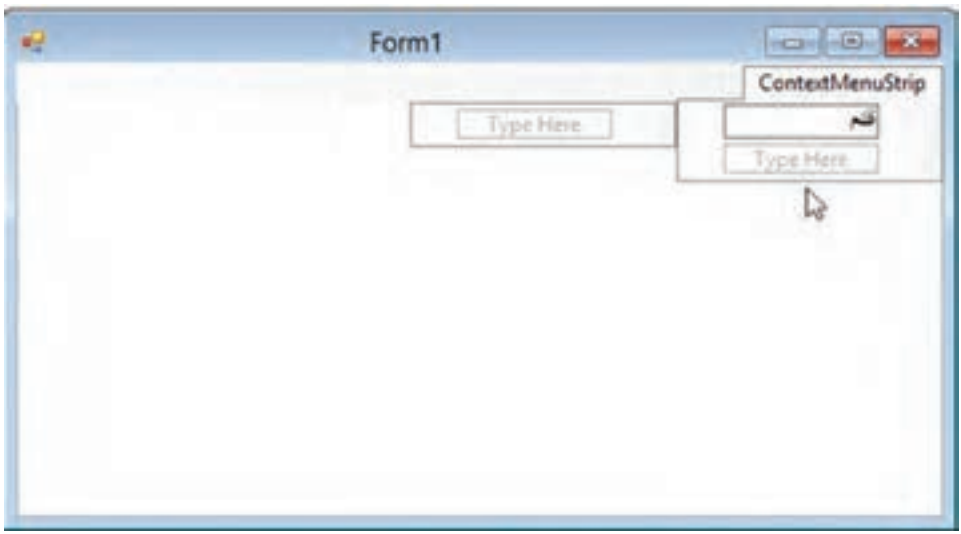
۳ــ درج کنترل: روی کنترل ContextMenuStrip در جعبه ابزار دوبار کلیک می‌کنیم تا در کلاس فرم تعریف شود.

۴ــ تنظیم ویژگی‌های منو

جدول ۴ــ۳ــ ویژگی‌های منو

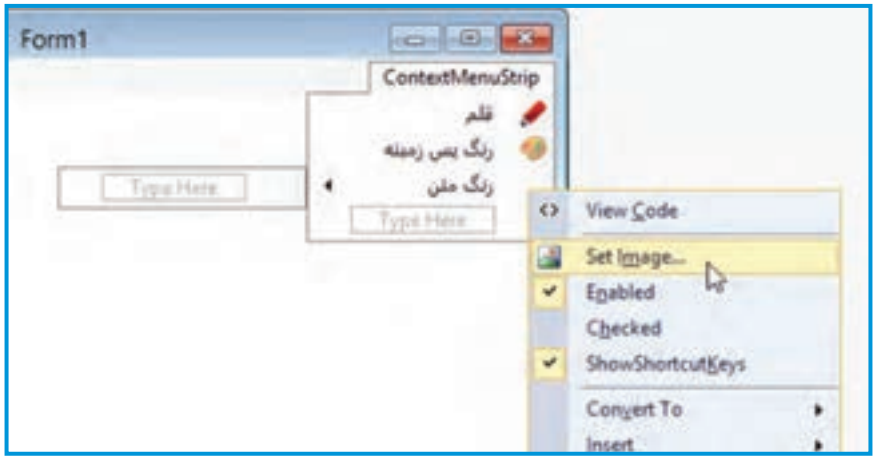
ContextMenuStrip	
ویژگی	مقدار
Name	RightClickMenu
RightToLeft	Yes

۵ــ درج گزینه‌های منو: مشابه آنچه که در بخش MenuStrip گفته شد می‌توانیم گزینه‌های منو را وارد کنیم. نام گزینه قلم را در پنجره properties برابر fontMenuItem قرار می‌دهیم. گزینه‌های «رنگ پس زمینه» و «رنگ متن» را نیز اضافه می‌کنیم و نام آنها را به ترتیب backColorMenuItem و textColorMenuItem قرار می‌دهیم. سپس برای هر گزینه تصویری را انتخاب می‌کنیم.



شکل ۱۷ــ۳ــ درج گزینه‌های منو

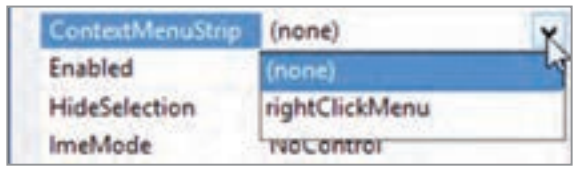
۶ــ نوشتن دستورات برنامه : اکنون نوبت نوشتن دستورهایی است که با کلیک گزینه‌های منو اجرا می‌شوند. برای آنکه متد رویداد کلیک هر گزینه را ایجاد کنیم کافی است که روی آن دوبار کلیک کنیم یا از پنجره properties و قسمت رویدادها، متد رویداد کلیک را ایجاد کنیم.



شکل ۱۸ــ۳ــ کدنویسی برای گزینه‌های منو

```
private void fontMenuItem_Click(object sender, EventArgs e)
{
    fontDialog1.ShowDialog();
    editor.Font = fontDialog1.Font;
}
private void backColorMenuItem_Click(object sender, EventArgs e)
{
    colorDialog1.ShowDialog();
    editor.BackColor = colorDialog1.Color;
}
private void textColorMenuItem_Click(object sender, EventArgs e)
{
     colorDialog1.ShowDialog();
     editor.ForeColor = colorDialog1.Color;
}
```

اکنون برنامه را اجرا کنید. خواهید دید که با کلیک راست روی کادر متنی، منوی طراحی‌شده نمایش داده نمی‌شود. برای اینکه منوی ما به عنوان منوی کلیک راست شناخته شود باید ویژگی ContextMenuStrip برای کادر متنی editor برابر با rightClickMenu شود.



شکل ۱۹ــ۳ــ ویژگی ContextMenuStrip

سؤال: آیا کلیدهای میان‌بر را می‌توان برای گزینه‌های منوی زمینه تعیین کرد؟ قابلیت انتخاب علامت را چطور؟ سؤال : آیا می‌شود به چند کنترل (مانند کادر متنی) یک منوی زمینه را تخصیص داد؟

خودآزمایی فصل سوم

الف) درستی یا نادرستی عبارات زیر را تعیین کنید :

۱ــ گزینه‌های منوی ایجاد شده با Insert standard Items را می‌توان تغییر داد.

۲ــ برای فعال سازی گزینه‌های منو فقط می‌توان از ماوس استفاده کرد.

۳ــ ویژگی Checked برای فعال یا غیر فعال کردن گزینه‌های منو به‌کار می‌رود.

۴ــ از خط جداکننده در طراحی منو برای دسته بندی گزینه‌ها استفاده می‌شود.

۵ــ در ویژوال استودیو امکان ایجاد منوی فارسی وجود ندارد.

ب) جاهای خالی را با عبارت مناسب پر کنید :

۶ــ برای ایجاد نوار منو از کنترل..... استفاده می‌شود.

۷ــ برای فعال سازی یک حرف از نام گزینه به عنوان کلید دسترسی، از علامت ..... قبل از حرف مورد نظر استفاده می‌کنیم.

۸ــ برای تعیین جهت منو از راست به چپ از ویژگی ..... استفاده می‌کنیم.

ج) به سؤالات زیر پاسخ دهید :

۹ــ در چه مواقعی از منوها استفاده می‌شود؟

۱۰ــ کاربرد کلید زمینه را برای گزینه‌های منو بنویسید.

۱۱ــ کاربرد کلید دسترسی برای گزینه منو چیست ؟

۱۲ــ تفاوت دو ویژگی Visible و Enabled را بنویسید.

تمرینات برنامه‌نویسی فصل سوم

۱ــ مثال ۵ ــ ۳ در کارگاه ۲ را طوری تغییر دهید که کادر متنی منبع، دارای یک منو با گزینه «فرستادن به مقصد» و کادر متنی مقصد دارای یک گزینه «گرفتن از منبع» باشد. با زدن هر یک از این گزینه‌ها، عملیات کپی از منبع به مقصد صورت پذیرد.

۲ــ یک پروژه ایجاد کنید و سه تصویر را به وسیلهٔ pictureBox طوری به فرم اضافه کنید که آبشاری روی هم قرار بگیرند. با کلیک راست روی هر تصویر باید منویی ظاهر شود که دارای دو گزینه باشد : «آوردن به جلو» و «فرستادن به عقب»

با انتخاب منوی آوردن به جلو باید تصویر روی سایر تصاویر قرار بگیرد و با انتخاب منوی «فرستادن به عقب» تصویر زیر سایر تصاویر قرار بگیرد.

راهنمایی : برای فرستادن یک کنترل مانند کادر تصویر از متد BringToFront و برای فرستادن به عقب از متد SendToFront استفاده می‌شود. بنابراین برای یک کادر تصویر با نام pictureBox1 داریم :

```
pictureBox1.BringToFront();
pictureBox1.SendToBack();
```

۳ــ برای برنامه بالا یک نوار منو با استفاده از کنترل MenuStrip و مطابق راهنمای صفحه بعد بسازید.

در این منو تصویرهایی که نمایش داده می‌شوند تیک خورده‌اند. با کلیک روی هر گزینه، اگر در حالت فعلی تیک خورده باشد، تصویر مربوط به آن پنهان می‌شود و تیک آن گزینه نیز برداشته می‌شود و اگر در حالت فعلی تیک ندارد برعکس این عمل انجام می‌شود.	
در این منو، تصویری که باید در جلوی همه باشد تیک خورده است. همچنین اگر یک گزینه که تیک ندارد کلیک شود، تصویر مربوط به آن به جلو می‌آید و خود آن گزینه تیک دار می‌شود و گزینه قبلی از حالت تیک دار خارج می‌شود (در کد، همه گزینه‌ها جز گزینه فعلی را از حالت Checked خارج می‌کنیم).	

فعالیت

تکمیل پروژه

برای تکمیل پروژه، به جز جستجو، نیاز به امکانات دیگری نیز داریم. **در این بخش** برای آنکه ظاهر فرم شلوغ نشود وهمچنین کاربرپسند باشد، رابط کاربری طراحی می‌کنیم که متداول است و در انواع واژه نامه‌ها دیده می‌شود. استفاده از منو سبب می‌شود دسترسی به امکانات از این طریق فراهم شود.

مرحله سوم :

۱ــ یک menuStrip به فرم اضافه شود و گزینه‌های آن به صورتی که در شکل زیر مشخص است تعیین شود و ویژگی name و Text گزینه‌ها مطابق جدول زیر تعیین شود.

ویژگی‌ها

ویژگی	مقدار
name	text
fileMenuItem	فایل
registerAndEditMenuItem	ثبت و ویرایش واژه
saveMenuItem	ذخیره جستجو
closeMenuItem	بستن

۲ــ همچنین یک منوی زمینه بـا عـنـوان «ذخیره جستجو» بـــرای کـــادر معنـی و تـوضیح، طراحـی و تعیین شود (شکل روبـه‌رو). نام این منو را saveMenuItem2 قرار می‌دهیم.

۳ــ پس از ایجاد منوهای بالا برای گزینه بستن ، کد متد EH رویداد کلیک آن، نوشته شود.

```
private void closeMenuItem_Click (Object sender, EventArgs e)
{
    this.Close ( );
}
```

۴ــ برای گزینه «ثبت و ویرایش واژه» متد EH رویداد کلیک را این‌گونه تعیین کنید.

```
private void registerAndEditMenuItem_Click (Object sender, EventArgs e)
{
    MessageBox.Show ("(: منتظرم ،تا ایجاد یک راه‌حل مناسب در فصل‌های بعدی");
}
```

اگر دکمه بستن زده شود، فرم بسته خواهد شد و اگر دکمه «ثبت و ویرایش» زده شود پیام آن نمایش داده می‌شود.

برای گزینه «ذخیره جستجو» نیز کد مورد نیاز را بنویسید تا همین پیام نمایش داده شود.

توجه : برای گزینه «ذخیره جستجو» در هر دو منو از یک متد EH استفاده کنید. در این برنامه، این متد را saveMenuItem_Click نام‌گذاری کنید.

واژگان و اصطلاحات انگلیسی فصل سوم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Backcolor	
۲	Checked	
۳	Context	
۴	Copy	
۵	Destination	
۶	Dialog	
۷	Enabled	
۸	Enter	
۹	Forecolor	
۱۰	Hot Key	
۱۱	Image	
۱۲	Insert	
۱۳	Item	
۱۴	Menu	
۱۵	Paste	
۱۶	Separatore	
۱۷	ShortcutKey	
۱۸	Source	
۱۹	Strip	
۲۰	Tool bar	
۲۱	Tools	
۲۲	Visible	

فصل چهارم

شیء و کلاس

در کتاب برنامه‌سازی ۲ با مفاهیم شیءگرایی[۱] به‌طور مختصر آشنا شدید. مفهوم شیء را شناختید و با کلاس‌های آماده آشنا شدید. در زبان‌های برنامه نویسی شیءگرا امکان تعریف ویژگی‌ها و رفتارهای اشیاء فراهم شده است. برنامه‌ای که به این زبان‌ها نوشته و اجرا می‌شود، در واقع از تعدادی شیء تشکیل شده است که با یکدیگر در ارتباط و تعامل هستند. در این فصل با کلاس و روش استفاده از آن در برنامه آشنا می‌شویم.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ کلاس و شیء را تعریف کند و برای آنها مثال بیاورد.

۲ــ متدها را بشناسد و تعریف نماید.

۳ــ تفاوت متد، فیلد و ویژگی را توضیح دهد.

۴ــ توصیف کننده‌های لازم را به‌کار بگیرد.

۱ــ Object Oriented

۱ــ۴ــ مفهوم شیء و کلاس

آیا تاکنون به نحوهٔ ساخت یک روبات فکر کرده‌اید؟ قطعاً با ایجاد یک طرح اولیه پیش از ساخت موافق هستید. در این طرح مشخص می‌کنید که روبات چه ویژگی‌هایی دارد، چه فعالیت‌هایی می‌تواند انجام دهد و چگونه این فعالیت‌ها را باید به سرانجام برساند. پس از ساختن این طرح، یک یا چندین نمونه از روبات را می‌توانید بسازید. در زبان برنامه‌نویسی طرح همان کلاس و شیء، نمونه‌ای از کلاس است.

سؤال: در مثال ساخت روبات، کلاس و شیء کدام است؟

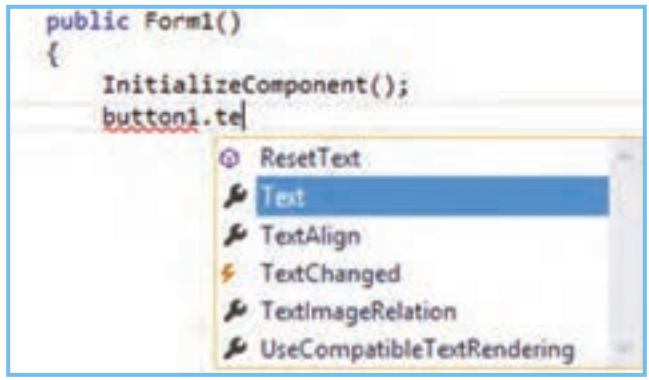
۲ــ۴ــ اشیاء آشنا

یک پروژه ایجاد کنید و روی فرم ایجاد شده، یک دکمه با نام button1 ایجاد کنید. تا همین لحظه شما دو شیء کاملاً آشنا دارید. فرم و دکمه در پروژهٔ فعلی. البته ناگفته نماند که در پس پرده، اشیاء بیشتری به‌وجود آمده‌اند! به نظر شما چه اشیایی؟!

به‌قسمت کد برنامه در فایل (Form1.cs) بروید و در متد مقداردهی اولیه فرم (InitializeComponent() عبارت **button1.** را بنویسید تا لیست هوشمند IntelliSense پدیدار شود. اعضای لیست را بررسی کنید.

سؤال: این موارد همگی متعلق به چه کنترل و شیء‌ای هستند؟

ویژگی Text را در لیست هوشمند پیدا و انتخاب کنید شکل (۱ــ۴).

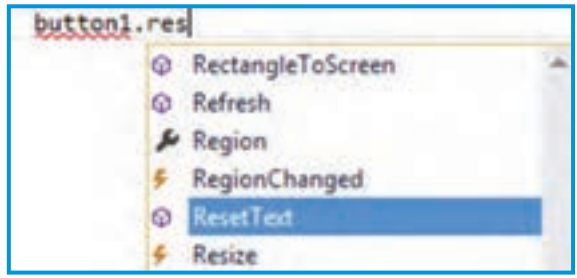


شکل ۱ــ۴

دستور زیر را بنویسید تا متن دکمه تغییر کند

```
button1.Text= "دکمه";
```

اکنون متد EH رویداد کلیک دکمه را با دو بار کلیک بر روی آن ایجاد کنید و در آن **button1** را بنویسید تا لیست هوشمند باز شود. این بار متد ResetText را پیدا کنید (شکل ۲ــ۴).



شکل ۲ــ۴

به آیکن متفاوت این گزینه با گزینه Text در لیست دقت کنید.

سؤال: چه تفاوتی بین این دوگزینه انتخابی می‌بینید؟

با انتخاب متد ResetText و کامل کردن آن داریم :

```
private void button1_Click(object sender, EventArgs e)
{
   button1.ResetText ( );
}
```

برنامه را اجرا کنید. خواهید دید که متن کنترل button1 برابر با «دکمه» است. دکمه را کلیک کنید خواهید دید که متن دکمه خالی می‌شود. آیا شما از چگونگی انجام این عملیات باخبر بودید؟ آیا برای آنکه با دکمه کار کنید نیاز است از کدهای درونی و شکل دهندهٔ دکمه و نحوه عملکرد آنها با خبر باشید؟

در انجام همین عملیاتِ ساده نکات خوبی مورد توجه قرار می‌گیرد :

- شیء button1 نمونه‌ای[۱] از یک کلاس است. این کلاس همان طرح اولیه برای ایجاد دکمه است و ویژگی‌های دکمه، عملیات آن و نحوهٔ انجام عملیات در آن ذکر شده است. شیء button1 براساس این طرح ایجاد شده است.
- Text یک ویژگی برای دکمه محسوب می‌شود. دقیقاً همان‌طور که می‌توانید بگویید چه متنی روی یک صفحه نوشته شده و یا متن روی صفحه را تغییر دهید، می‌توانید از ویژگی Text دکمه برای خواندن متن دکمه یا تغییر آن استفاده کنید.

۱ــ Instance

- ResetText یک متد است؛ بنابراین بیانگر عملیات یا رفتاری است. این متد، متنِ روی دکمه را پاک می‌کند. هنگام استفاده از این متد نیازی نیست که از چگونگی انجام این عمل مطلع باشیم که به وسیلهٔ همان کدهای درونیِ متد صورت می‌گیرد.
- شیء button1 از ویژگی‌ها و متدهای مختلف تشکیل شده است[1] که همه در طرح اولیه تعریف شده است.
- شیء button1 یک هویت مستقل دارد و برای کار کردن نیاز به کدهای آن نداریم. تنها چیزی که لازم داریم این است که بدانیم این شیء دارای چه ویژگی‌ها و متدهایی است وهرکدام چه کاربردی دارند.

۳ــ۴ــ اشیاء چگونه ساخته می‌شوند؟

همان‌طورکه بیان شد، شیء بر اساس کلاس ساخته می‌شود. در واقع شیء، نمونه ساخته شده بر اساس نقشه و مدلی است که کلاس مشخص می‌کند. اگر شما طرح یک روبات را داشته باشید به‌معنای آن نیست که آن روبات را اکنون در دست دارید. ابتدا باید هزینه کنید و قطعات آن را تهیه کرده و بر اساس طرح، سر هم کنید. ساختن یک شیء هم برای سیستم، نیاز به هزینه دارد. هر شیء که ساخته می‌شود به میزان مورد نیاز، فضایی از Ram به آن اختصاص داده می‌شود.

دستور ایجاد یک شیء جدید از کلاس در حالت کلی به صورت زیر است :

new نام کلاس ( )

به پروژهٔ خود باز می‌گردیم. ابتدا می‌خواهیم ببینیم کلاس Form1 کجا ایجاد شده است. فایل کد Form1.cs را باز کنید. در کد این فایل همان‌طور که بارها دیده‌ایم و حتی نام برده‌ایم کلاس Form1 تعریف شده است.

```
public partial class Form1 : Form
{
    ادامهٔ برنامه
}
```

ایجاد کلاس Form1 از روی کلاس Form

۱ــ البته همان‌طور که مشاهده می‌شود ما رویدادهایی نیز داریم که یک نحوهٔ خاص از به‌کارگیری و اجرای متدها است.

تعریف کلاس Form1 از اینجا شروع می‌شود . البته این یک بخش از تعریف کلاس Form1 است.

قسمت اصلی تعریف کلاس فرم ما، class Form1 می‌باشد. کلمات public و partial توصیف کننده می‌باشند که در ادامهٔ فصل توضیح بیشتری دربارهٔ توصیف کننده public داده خواهد شد. همچنین در این تعریف، قسمتی که با کادر مشکی مشخص شده است، به این معناست که کلاس Form1 از روی کلاس Form ساخته شده است و به نحوی کلاس Form حق پدری بر گردن آن دارد!

سؤال: کلاس Form کجا تعریف شده است؟

در این بخش از تعریف Form1، شما تاکنون کدهای زیادی نوشته‌اید و دیده‌اید. بنابراین به خوبی با آن آشنا هستید و می‌دانید معمولاً چه کدهایی در آن نوشته می‌شود. اما بخش دیگری هم هست که آن را حتماً دیده‌اید. فایل Form1.Designer.cs را در ویژوال استودیو باز کنید. این کدها نیز بخشی از کلاس Form1 هستند.

اما سؤال اینجاست اکنون که کلاس Form1 ایجاد شده است پس کجا شیءای از آن ساخته شده که ما آن را به عنوان یک فرم می‌توانیم در هنگام اجرا ببینیم؟ فایل Program.cs را باز کنید و کد زیر را پیدا کنید :

```
Application.Run(new Form1());
```

ایجاد شیء از کلاس Form1

سؤال: قسمت مشخص شده با کد new Form() چه چیزی را نشان می‌دهد؟

اکنون با کنجکاوی در شیء button1 می‌خواهیم بدانیم کلاس مربوط به این شیء چیست و در کجا این شیء به‌وجود آمده است؟

دوباره فایل Form1.Designer.cs را باز کنید. این دو خط را پیدا کنید :

```
private System.Windows.Forms.Button button1;
```

تعیین نوع شیء button1

```
this.button1 = new System.Windows.Forms.Button();
```

ایجاد شیء button1

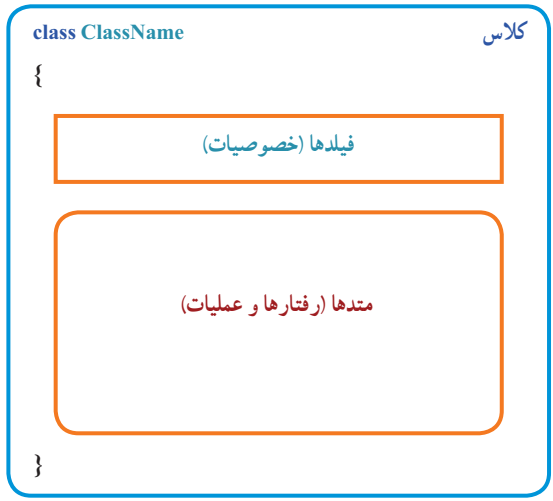
سؤال: شیء button1 از چه کلاسی می‌باشد؟

۴ــ۴ــ شیء گرایی

جهان پیرامون ما پر از اشیاء گوناگون و متنوعی است که هریک ویژگی‌ها و رفتارهای مخصوص به خود دارند. در جهان عینی، ما معمولاً هر جسم فیزیکی را شیء قلمداد می‌کنیم. در برنامه‌نویسی این اجسام و در کل هر موجودیتی که دارای ویژگی‌ها و فعالیت‌های خاص خود باشد به صورت یک شیء می‌تواند طرح ریزی و استفاده شود. از جمله زبان‌هایی که امکان تعریف شیء و برنامه‌نویسی بر همین اساس را مهیا کرده است زبان C# می‌باشد. استفاده از مفاهیم شیء گرایی، بر قدرت برنامه می‌افزاید و طراحی آن را حرفه‌ای‌تر می‌نماید.

۵ــ۴ــ تعریف و به‌کارگیری متد

خصوصیات و وضعیت یک شیء به وسیله فیلدها و رفتارهای اشیاء در قالب متدها تعریف می‌گردند. بنابراین محل و مکان تعریف فیلدها و متدهای یک شیء در داخل یک کلاس است :



شما تا پیش از این با ویژگی‌ها و متدهای اشیاء و کلاس‌های مختلفی کار کرده اید. اما برای ایجاد آنها در کلاس، نیاز به اطلاعات و تمرین بیشتری است. بنابراین ابتدا به تمرین ایجاد متدها می‌پردازیم و بعد گام به گام پیش می‌رویم.

می‌دانید که متد مجموعه‌ای از دستورات است که عمل خاصی را انجام می‌دهد. هر متد می‌تواند تعدادی ورودی داشته باشد و حداکثر یک مقدار برگشتی یا خروجی نیز داشته باشد. برای آنکه یک متد را ایجاد کنیم آن را داخل یک کلاس می‌نویسیم. نحوهٔ نوشتن یک متد به صورت زیر است.

```
توصیف‌کننده    نوع خروجی    نام متد (ورودی‌ها)
{
    دستورات
}
```

ورودی‌ها به شکل زیر مشخص می‌شوند :

```
نوع ورودی ۱ نام ورودی ۱ , نوع ورودی ۲ نام ورودی ۲ , ...
```

تعداد ورودی‌ها بستگی به شرایط دارد . گاهی ممکن است یک متد نیازی به ورودی هم نداشته باشد.

برای مثال فرض کنید می‌خواهیم متدی داشته باشیم که نام و نام خانوادگی را بگیرد و نام کامل را برگرداند. همان‌طور که از صورت مثال مشخص است دو ورودی از نوع string لازم داریم و نوع خروجی نیز string است.

```
private string GetFullName(string name,string family)
{
 return name +" "+ family ;
}
```

سؤال: کدام string نوع خروجی را مشخص می‌کند؟

در این مثال از کلمهٔ کلیدی private برای توصیف‌کنندهٔ متد استفاده شده است که در ادامه بحث کلاس به کاربرد آن نیز خواهیم رسید. کلمهٔ کلیدی return برای برگشت دادن یک مقدار به کار می‌رود. در این متد می‌بینیم که رشته‌های متنی نام و نام خانوادگی به همراه یک کاراکتر فاصله در میان آنها، الحاق شده‌اند و رشته متنی حاصل، برگشت داده شده است.

سؤال: در فراخوانی متد به شکل زیر، چه مقداری در متغیر fname قرار خواهد گرفت؟

```
string fname = GetFullName("علی","محمدی")
```

کار در کارگاه ۱: بازی دقت و قدرت حافظه

می‌خواهیم یک بازی دو نفره طراحی کنیم که نفر اول یک عدد چند رقمی‌وارد کند و دکمه شروع را بزند و بازیکن دوم اقدام به نوشتن همان رقم به صورت بر عکس نماید. با زدن دکمه پایان از طرف بازیکن دوم بازی به اتمام رسیده و نتیجه اعلام شود.

الگوریتم یا روش انجام کار : با زدن دکمه شروع ، عدد وارد شده ذخیره می‌شود و متن کادر متنی پاک می‌شود. با زدن دکمه پایان عدد وارد شده توسط کاربر با عکس رشته متنی ذخیره شده مقایسه می‌شود و پیام مناسب پیروزی یا شکست نمایش داده می‌شود. برای به‌دست آوردن عکس رشته متنی یک متد می‌نویسیم.

شکل ۳ــ۴ــ فرم برنامه

۱ــ ایجاد پروژه و فرم برنامه : پروژه جدید WFA را ایجاد می‌کنیم و اندازه فرم را به مقدار مناسب تعیین می نماییم.

۲ــ وارد کردن کنترل‌ها به فرم : در این فرم ما به کنترل‌های زیر نیاز داریم :

جدول ۱ــ۴ــ ویژگی‌های کادر متن

TextBox		
ویژگی		مقدار
Name		input
Text		
Font	size	14

جدول ۲ــ۴ــ ویژگی‌های برچسب

Label	
ویژگی	مقدار
Name	label1
Text	عدد مورد نظرتان را وارد کنید
AutoSize	True
RightToLeft	Yes

جدول ۳ــ۴ــ ویژگی‌های دکمه‌ها

Button		
ویژگی	مقدار	مقدار
Name	start	finish
Text	شروع	پایان
Enabled	True	False

۳ــ *تعریف متد:* ابتدا یک متد در فرم می‌سازیم که یک رشته متنی را بگیرد و عکس آن را تحویل بدهد. توجه داشته باشید که گرچه ما عدد وارد می‌کنیم اما در واقع با رشته متنی وارد شده کار می‌کنیم و هر چه که باشد آن را عکس می‌نماییم.

```
private string Reverse (string str)
{
    string result = "";
   for (int c = str.Length−1; c >= 0; c− −)
   {
      result += str [c];
   }
  return result;
}
```

در این متد حرف به حرف از انتهای رشته متنی ورودی str تا ابتدای آن خوانده می‌شود و به رشته متنی result که در ابتدای کار خالی است، اضافه می‌شود. در پایان نیز رشته متنی result برگشت داده می‌شود.

اکنون متد EH رویداد کلیک دکمه‌ها را می‌نویسیم. برای دکمه start کافی است که با زدن آن دکمه finish فعال شود و خود دکمه start غیر فعال شود. همچنین نیاز است که رشته متنی وارد شده قبل از پاک شدن، در یک متغیّر ذخیره شود. بنابراین ما یک متغیر در داخل کلاس فرم ایجاد می‌کنیم تا برای متدهای دیگر قابل شناسایی باشد و مقدار خود را نیز حفظ نماید.

```
public partial class Form 2 :  Form
{
    string goal;
   ادامه برنامه
}
```

۴ــ تعریف متد *EH رویدادها* : متد EH رویداد کلیک دکمه start را این‌چنین ایجاد می‌نماییم :

```
private void start_Click (object sender, EventArgs e)
{
    finish.Enabled = true;
    start.Enabled = false;
    goal = input.Text;
   input.Text = "";
}
```

متد EH رویداد دکمه پایان را نیز به صورت زیر ایجاد می‌کنیم :

```
private void finish_Click (object sender, EventArgs e)
{
    finish.Enabled = false;
    start.Enabled = true;
    if (Reverse(goal) == input.Text);
      MessageBox.Show("آفرین! جواب درست را وارد کردید");
   else
     MessageBox.Show("متأسفانه جواب شما درست نبود!");
}
```

با کلیک دکمه پایان، خود دکمه غیر فعال می‌شود. دکمه شروع فعال می‌شود و عکس رشته متنی اصلی توسط متد Reverse بدست آمده ، با رشته متنی ورودی توسط بازیکن دوم مقایسه شده و پیام مناسب داده می‌شود.

توسعه و بهبود برنامه

می‌خواهیم دکمه‌ای در فرم بگذاریم که اگر کلیک شود متن ابتدایی و عکس آن را به صورت پیام نشان دهد.

شکل ۴ــ۴ــ نمایش جواب

دکمه‌ای با مشخصات زیر به فرم اضافه می‌کنیم.

جدول ۴ــ۴ــ ویژگی‌های دکمه

شیء Button	
ویژگی	مقدار
Name	showAnswer
Text	نمایش جواب
Enabled	True

متد رویداد کلیک آن را به صورت زیر می‌نویسیم.

```
private void showAnswer_Click(object sender, EventArgs e)
{
MessageBox.Show("متن وارد شده برابر با " + goal + " و عکس آن برابر با " + Reverse(goal)+ "می‌باشد");
```

همان‌طور که می‌بینید از مزایای متد آن است که ما یک‌بار آن را تعریف می‌کنیم و بارها آن را می‌توانیم استفاده کنیم. همچنین اگر بخواهیم آن را در برنامه ای دیگر استفاده کنیم به راحتی می‌توانیم متد را در برنامه مورد نظر کپی کنیم و به کار ببندیم.

توسعه و بهبود برنامه

برای اطمینان از ورود اعداد در کادر متن از یکی از روش‌های زیر می‌توان استفاده کرد :

۱ــ برای آنکه فقط با اعداد کار شود ابتدا بررسی کنید که آیا رشته متنی وارد شده یک عدد است یا خیر؟

۲ــ از طریق کدنویسی کادر متنی را طوری تغییر دهید که فقط عدد وارد شود.

۶ــ۴ــ استفاده از کلاس و شیء

مثال ۱ــ۴ : یک ساعت را به عنوان یک شیء در نظر بگیرید. می خواهیم ساده ترین ویژگی‌ها و عملیات مربوط به یک ساعت را مشخص و در یک کلاس تعریف کنیم.

شکل ۵ــ۴

الگوریتم یا روش انجام کار : برای نمایش ساعت، از سه ویژگی به نام ساعت، دقیقه و ثانیه استفاده می‌شود. مدل ساعت (عقربه‌ای یا دیجیتال)، رنگ، شکل ظاهری، جنس و قیمت از ویژگی‌های دیگر ساعت هستند اما برای نمایش زمان فقط به سه ویژگیِ اول نیاز داریم.

سؤال: به جز نمایش زمان و تنظیم ساعت که از جمله عملیاتی است که روی ساعت تعریف می‌شود، چه عملیات دیگری به فکر شما می‌رسد؟

در این مثال، نام کلاس را Clock انتخاب می‌کنیم. روش نوشتن نام کلاس، روش پاسکال و روش نوشتن نام فیلدها، روش کوهان شتری است (قطعه برنامه ۱ــ۶).

```
Class Clock
{
                                    فیلدها
byte hour, minute, second;

 متدها
public void SetClock (byte h, byte m, byte s)
{
   hour = h;
   minute = m;
   second = s;
}
public  void ShowClock ( )
{
   MessageBox.Show(hour.ToString ( ) +" : " + minute.ToString( )
+ " " + second.ToString ( ));
}
```

قطعه برنامه ۱ ــ۶ــ کلاس ساعت

سه متغیر از نوع byte برای نگهداری مقدار ساعت، دقیقه و ثانیه استفاده شده است. متغیرهای hour، minute و second فیلدهای کلاس نامیده می‌شوند. متد ()SetClock برای تنظیم ساعت استفاده شده که دارای سه پارامتر است. همان‌طور که از خط عنوان متد دیده می‌شود، سه عدد به‌عنوان پارامتر به این متد وارد می‌شود و با اجرای متد مذکور، ویژگی‌های شیء ساعت، مقداردهی جدید شده و تنظیم می‌شود. به نوع خروجی این متد توجه کنید! کلمه کلیدی void می‌گوید که متد، هیچ مقداری را برنمی‌گرداند. همان‌طور که می‌بینید در متد، دستور return استفاده نشده است و مقداری بازگشت داده نمی‌شود[1]

۱ــ در متدهای که خروجی ندارند دستور ;return برای خروج از متد می‌تواند به‌کار رود.

متد ShowClock() وظیفه نمایش ساعت بر روی صفحه را به‌عهده دارد. این متد خروجی ندارد.

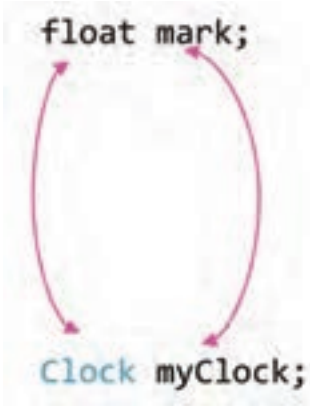
سؤال: از چه دستوری برای نمایش ساعت استفاده شده است؟

سؤال: آیا کلاسی که ایجاد می‌کنیم باید در کلاس فرم باشد؟

کلاس می‌تواند در هر جای دلخواه غیر از داخل متدها تعریف شود :

نکته

توجه شود، اگر کلاسی را بیرون از کلاس فرم اما در همان فایل ایجاد می‌کنید آن‌را بعد از کلاس فرم بنویسید چون برای محیط طراحی فرم اشکال به‌وجود می‌آید. هرچند در زمان اجرا خطایی رخ نخواهد داد.



شکل ۶ــ۴

به‌خاطر دارید که برای تعریف یک متغیر ابتدا نوع آن و سپس نام متغیر را می‌نوشتیم. مثلاً برای تعریف متغیری از نوع اعشاری دستور float mark را به کار می‌بردیم. این در دستور، متغیر mark از نوع float است.

برای ایجاد شیء ساعت از نوعِ کلاسِ Clock هم از همان شیوه استفاده می‌کنیم :

```
Clock    myClock;
```

در دستور بالا شیء myClock از نوع Clock است.

اما تعریف بالا هنوز کامل نیست و فقط مرحلهٔ اوّلِ تعریف است. در زبان سی شارپ برای تعریف یک شیء از یک کلاس، از عملگر new استفاده می‌کنیم. به وسیلهٔ عملگر new در دستور زیر یک شیء ساخته می‌شود اما این شیء نامی ندارد که هنگام استفاده، آن را به کار ببریم.

```
new Clock();
```

در دستور بالا، شیء ایجاد شده وحافظه‌ای برای آن تخصیص داده می‌شود اما چگونه می‌توان از آن استفاده نمود؟

برای اینکه شیء موردنظر نامی‌داشته باشد تا بتوان از این نام در برنامه استفاده کرد، دستور بالا را به صورت زیر تصحیح می‌کنیم :

```
myClock = new Clock();
```

نکته

بنابراین برای استفاده از شیء ایجاد شده در دستورهای بعدی می‌توانیم به‌صورت زیر عمل نماییم

```
Clock myClock;                    ← مرحلهٔ اول
myClock = new Clock();            ← مرحلهٔ دوم
```

می توانید دو خط فوق را به صورت زیر خلاصه کنید :

```
Clock myClock = new Clock();
```

۷ــ۴ــ دسترسی به فیلدها و متدهای یک شیء

پس از ایجاد شیء می‌توانید به فیلدها و متدهای شیء دسترسی داشته باشید و آنها را فراخوانی کنید. برای دسترسی به اجزا و اعضای یک شیء، کافی است نام شیء و سپس نام عضو (فیلد یا نام متد) را به دنبال آن ذکر کنیم که با علامت نقطه از یکدیگر جدا می‌شوند.

نام عضو.نام شیء

برای مثال برای تنظیم ساعت، متد SetClock() و برای نمایش ساعت از فراخوانی متد ShowClock() استفاده می‌کنیم.

کار در کارگاه۲

مطابق شکل ۷ــ۴ فرمی با دو دکمه برای شیء ساعت است، آن را طراحی کنید و کدهای موردنیاز را بنویسید.

شکل ۷ــ۴

الگوریتم یا روش انجام کار :

۱ــ *ایجاد پروژه :* یک پروژهٔ جدید WFA ایجاد می‌نماییم.

۲ــ *اضافه کردن کنترل‌ها :* ابتدا دو groupBox ایجاد می‌کنیم و مقدار Text آنها را خالی می‌کنیم. سپس طبق شکل ۷ــ۴ به این دو groupBox، کنترل‌های زیر را اضافه می‌کنیم.

جدول ۵ــ۴ــ ویژگی‌های دکمه

Button		
ویژگی	**مقدار**	**مقدار**
Name	setClockBtn	showClockBtn
Text	تنظیم ساعت	نمایش ساعت

جدول ۶ــ۴ــ ویژگی‌های کنترل عددی افزایشی کاهشی

NumericUpDown			
ویژگی	**مقدار**	**مقدار**	**مقدار**
Name	hourBox	minuteBox	secondBox
Minimum	۰	۰	۰
Maximum	۲۳	۵۹	۵۹

۳ــ *تعریف کلاس و شیء ساعت :* کلاس Clock که در قطعه برنامه ۱ــ۶ آمده را بعد از کلاس فرم می‌نویسیم. در درون کلاس فرم، متغیر myClock را از جنس Clock تعریف می‌کنیم.

```
public partial class Form1: Form
{
 Clock myClock = new Clock();
```

۴ــ اضافه کردن متد EH رویدادها : متدEH رویداد کلیک دکمه setClockBtn را به صورت زیر می‌نویسیم :

```
private void setClockBtn_Click(object sender, EventArgs e)
{
    myClock.SetClock((byte)hourBox.Value,(byte)minutBox.Value,(byte)
    secondBox.Value);
}
```

سؤال: علت آنکه ما برای تبدیل نوع، از عبارت byte استفاده کرده‌ایم چیست؟

متدEH رویداد کلیک دکمه showClockBtn را نیز به‌صورت زیر می‌نویسیم :

```
private void showClockBtn_Click(object sender, EventArgs e)
{
    myClock.ShowClock();
}
```

در نتیجهٔ این طراحی، اگر کادرهای عددی را به‌صورت ۰ :۳۱ :۹ تنظیم کنیم و دکمه تنظیم ساعت را بزنیم و سپس دکمهٔ نمایش ساعت را کلیک کنیم کادر زیر نمایش داده خواهد شد (شکل ۸ــ۴).

شکل ۸ــ۴

۸ــ۴ــ توصیف کننده‌ها

در کلاس Clock توصیف‌کننده متد ShowClock را از public به private تغییر دهید. سپس ماوس را روی کد فراخوانی متد ShowClock قراردهید، شرح متد به‌صورت شکل ۹ــ۴ ظاهر خواهدشد.

```
private void showClockBtn_Click(object sender, EventArgs e)
{
    myClock.ShowClock();
}
```

```
void Clock.ShowClock()

Error:
'WindowsFormsApplication3.Clock.ShowClock()' is inaccessible due to its protection level
```

شکل ۹ــ۴ــ خطای عدم دسترسی به متد فراخوانی شده

خط زیر متد نشان دهندهٔ خطا است. وقتی ماوس را روی فراخوانی متد می‌بریم راهنما به ما می‌گوید این متد به دلیل سطح محافظتی غیرقابل دسترس است! حتی اگر بخواهیم دوباره آن را بنویسیم خواهیم دید که دیگر این متد در لیست هوشمند دیده نمی‌شود.

```
myClock.;
}
```

- Equals
- GetHashCode
- GetType
- SetClock
- ToString

شکل ۱۰ــ۴ــ لیست هوشمند شیء myclock

تفاوت دو توصیف‌کننده را ببینید :

جدول ۷ــ۴

توصیف کننده	توضیح	مثال
private	دسترسی از بیرون به متغیر یا متدی که با این توصیف‌کننده مشخص شده، وجود ندارد و در واقع فقط داخل کلاس (شیء) قابل استفاده است. اگر متد یا متغیری توصیف کننده نداشته باشد به‌صورت پیش فرض private خواهد بود.	مانند متغیرهای hour، minute و second که private هستند و می‌توان توسط دستورات داخلی کلاس Clock به آنها دسترسی داشت اما نمی‌توان از بیرون به این متغیرها دست پیدا کرد.
public	به یک متد یا متغیر از شیءای که در کلاس به‌صورت public تعریف شده است می‌توان از بیرون کلاس، دسترسی داشت.	مانند متدهای ShowClock و SetClock در کلاس Clock

کاردرکارگاه ۳

مثال ۲ــ۴ : می‌خواهیم برای یک بازی جنگی فضایی[۱] که در آن سفینه‌های[۲] مختلفی وجود دارد و با هم در نبرد[۳] هستند، کلاسی برای سفینه تعریف کنیم.

پروژه‌ای از نوع WFA ایجادکرده (شکل۱۱ــ۴) و روی فرم یک دکمه با ویژگی‌های مشخص‌شده در جدول ۸ــ۴ قرار دهید.

شکل ۱۱ــ۴

۱ــ Space Wars
۲ــ Spaceships
۳ــ Battle

جدول ۸ــ۴ــ ویژگی دکمه

شیء Button	
مقدار	ویژگی
ShowButton	Name
show	Text

شکل ۱۲ ــ۴ــ فرم مثال ۲ــ۴

الگوریتم یا روش انجام کار : یک سفینهٔ فضایی ویژگی‌های مختلفی دارد، که بسته به نوع و سناریوی بازی، بعضی از آنها اهمیت پیدا می‌کند. در یک سفینه جنگی، ویژگی‌ها قدرت آتش[۱] و اندازهٔ مهمات[۲] و در یک سفینهٔ باری[۳] ویژگی ظرفیت باربری[۴] اهمیت دارد. ویژگی‌های مشترکی نیز در هر دو نوع سفینه مانند سرعت[۵]، سپردفاعی[۶] و اندازه سوخت[۷] وجود دارد. برای سادگی کار، فعلاً ویژگی مشترک بین سفینه‌ها را در یک کلاس تعریف می‌کنیم. بنابراین سه فیلد برای ویژگی‌های اندازه سرعت سفینه، قدرت سپر دفاعی و اندازه سوخت به‌صورت عدد صحیح بین ۰ تا ۱۰۰ (به عنوان درصد) تعریف می‌کنیم.

بنابراین تعریف فیلدهای کلاس سفینه چنین خواهد بود :

```
public class SpaceShip
{
  public int fuel;
  public int shield;
  public int speed;
}
```

۱ــ Fire Rate

۲ــ Armor

۳ــ Cargo Spaceship

۴ــ Storage

۵ــ Speed

۶ــ Shield

۷ــ Fuel

علاوه بر تعریف فیلدها، نیاز به انجام عملیات بر روی سفینه است. یک عمل معمول در سفینه، پر کردن[1] سوخت سفینه و یا کامل کردن سلامتی[2] سفینه است. با توجه به تعریف کلاس فوق، سه روش در اختیار داریم :

۱ــ استفاده از دستور انتساب و دسترسی به فیلدهای شیء

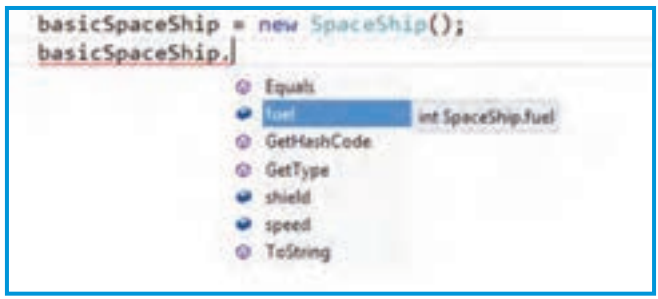
۲ــ استفاده از یک متد برای تغییر دادن مقدار فیلدها

۳ــ استفاده از ویژگی (Property) (راه حل استاندارد و بهینه در C#)

برای استفاده از هر یک از سه روش فوق ابتدا باید شیء از نوع کلاس سفینه ساخت. به دلیل آنکه ما می‌خواهیم شیء سفینه را در طول برنامه نگهداری و استفاده کنیم، در کلاس فرم متغیری از نوع کلاس سفینه تعریف کرده و آن را ایجاد می‌کنیم.

```
public partiral class Form1 : Form
{
    SpaceShip basicSpaceShip;  ⟵ تعریف متغیر
    public Form1 ( )
    {
        InitializeComponent ( );
        basicSpaceShip = new SpaceShip ( );⟵ ایجاد شیء سفینه
    }
```

روش اول ــ استفاده از دستور انتساب و دسترسی به فیلدهای شیء : با توجه به اینکه فیلدهای کلاس از نوع public تعریف شده اند، در خارج از کلاس سفینه نیز می‌توان به آنها دسترسی داشت و آنها را مقداردهی کرد.



شکل ۱۳ــ۴ــ لیست هوشمند شیء سفینه

۱ــ Fill ۲ــ Health

همان طور که در شکل ۱۳ــ۴ مشاهده می‌کنید در لیست ظاهر شده در IntelliSense نام هر سه فیلد fuel, shield, speed وجود دارد. علاوه بر سه فیلد، نام چهار متد نیز دیده می‌شود. آیا این متدها را ما تعریف کرده‌ایم؟ این متدها برای تمام اشیاء وجود دارند. علت وجود آنها را با پیشرفت در حوزهٔ برنامه‌نویسی و تحقیق درخواهید یافت.

اکنون دستورات پرکردن سوخت، سرعت و سپر دفاعیِ کاملِ سفینه را با استفاده از دستورات انتساب بر روی فیلدها مشاهده می‌کنید.

```
namespace classSpaceShip
{
  public partial class Form1 : Form
  {
        SpaceShip basicSpaceShip;
        public Form1( )
        {
            InitializeComponent ( );
            basicSpaceShip = new SpaceShip ( );
            basicSpaceShip. fuel = 100;
            basicSpaceShip. speed = 100;
            basicSpaceShip. shield = 100;
        }
  }
}
class SpaceShip
{
  public int fuel;
  public int speed;
  public int shield;
}
```

مقداردهی فیلدهای شیء سفینه

برنامه ۲ــ۶ــ مقداردهی فیلدها در مثال ۲ــ۶

با اجرای برنامه ۲ــ۶، در داخل فیلدهای شیء basicSpaceShip عدد ۱۰۰ به منزله مقدار صد درصد آن ویژگی‌ها قرار می‌گیرد.

برای اطلاع از مقدار فیلدها،در متد EH رویداد کلیکِ دکمه showButton دستوری بنویسید که مقدار فیلدها را در پنجره پیام نمایش دهد.(شکل ۱۴ــ۴)

```
private void showButton_Click (object sender, EventArgs e)
{
  string s= "speed: " + basicSpaceShip1.Speed+
            "\n fuel: " + basicSpaceShip1. Fuel+
            "\n shield: " + basicSpaceShip1|.Shield;
  MessageBox.Show(s, "مقدار خصوصیات سفینه");
}
```

در صورت کلیک دکمه showButton پنجره پیام مانند شکل۱۴ــ۴ ظاهر می‌شود.

شکل ۱۴ ــ۴

روش دوم ــ استفاده از یک متد برای تغییر دادن مقدار فیلدها : روش انتساب و تغییر دادن مقدار فیلدها، چندان مناسب نیست. ممکن است در جای دیگری از کد برنامه، دستوری بخواهد از این کلاس استفاده کند و برای پر کردن سوخت سفینه، به جای اعداد ۰ تا ۱۰۰ از محدودهٔ دیگری استفاده کند. مثلاً برای بالا بردن سوخت سفینه عدد ۵۰۰ یا ۱۰۰۰ و حتی بالاتر را استفاده کند که در این صورت ساختار برنامه را به‌هم می‌زند. برای این کار بهتر است که دسترسی به فیلدهای یک کلاس را محدود کنیم و اجازه ندهیم برنامه‌ای خارج از کلاس بتواند به آنها دسترسی داشته باشد. برای این منظور دسترسی آنها را از نوع private تعریف می‌کنیم که حوزهٔ شناخت آنها فقط محدود به کلاس شود. در این صورت لازم است متدهایی را برای تغییر دادن مقدار فیلدها پیش بینی کنیم تا عددی که می‌خواهد در فیلد قرار گیرد را کنترل کرده و مقدار مناسب را در آن فیلد قرار دهند. نوع دسترسی

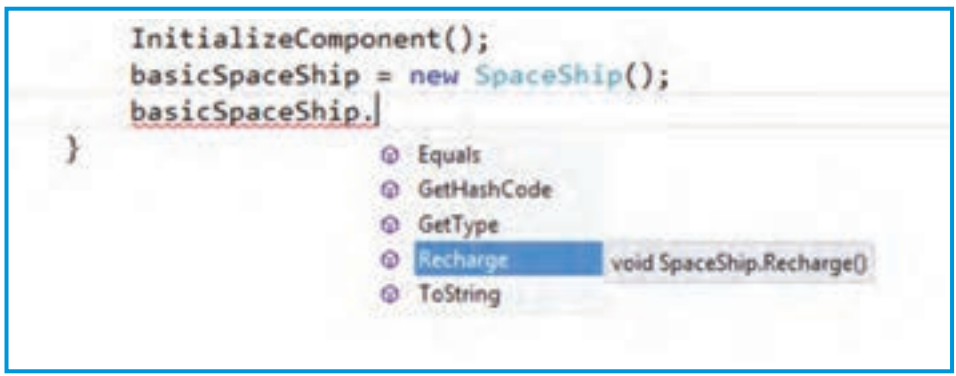
این متدها معمولاً باید public باشد تا برنامه‌های دیگر بتوانند از طریق فراخوانیِ این متدها، فیلدها را به طور صحیح مقداردهی نمایند.

این متدها را در کجای برنامه تعریف کنیم و یا تایپ کنیم؟ این متدها همان رفتارها و عملیات شیء مورد نظر ما هستند؛ بنابراین متدهای مربوطه در داخل کلاس قرار می‌گیرند.

```
public class SpaceShip
{
    private int fuel;
    private int shield;
    private int speed;
    public void Recharge ( )
    {
        fuel = 100;
        shield = 100;
        speed = 100;
    }
}
```

تعریف متد در کلاس

هنگامی که شیء از نوع کلاس سفینه ایجاد می‌شود و می‌خواهید به اجزای شیء دسترسی داشته باشید، به لیست IntelliSense دقت کنید که دیگر خبری از سه فیلد نیست. چرا؟



شکل ۱۵ ــ ۴

با توجه به مطالب گفته شده و تغییری که در کلاس سفینه فضایی دادیم، برنامه چنین خواهد شد :

```
namespace classSpaceShip
{
    public partial class Form1 : Form
    {
        SpaceShip basicSpaceShip;
        public Form1( )
        {
            InitializeComponent ( );
            basicSpaceShip = new SpaceShip ( );
            basicSpaceShip.Recharge ( );
        }
    }
}
class SpaceShip
{
    private int fuel;
    private int speed;
    private int shield;
    public void Recharge ( )
    {
        fuel = 100;
        speed = 100;
        shield = 100;
    }
}
```

همچنین برای اطلاع از مقدار یک فیلد، نیز می‌توانید متدی بنویسید که خروجی آن مقدار فیلد را به برنامهٔ اصلی بازگرداند. در تعریف زیر، سه متد برای اطلاع از مقدار فیلدهای شیء نوشته شده است :

```
public class SpaceShip
{
    private int fuel;
    private int shield;
    private int speed;
    public void Recharge ( )
    {
        fuel = 100;
        shield = 100;
        speed = 100;
    }
    public int GetFuel ( )
    {
        return fuel;
    }
    public int GetShield ( )
    {
        return Shield;
    }
    public int GetSpeed ( )
    {
        return Speed;
    }
}
```

بنابراین در این روش، اگر بخواهید هر فیلد را به‌صورت مجزا مقداردهی کنید یا مقدارش را بخوانید باید برای هر فیلد دو متد بنویسید : یک متد برای انتساب مقدار به فیلد و متد دیگری برای اطلاع از مقدار فیلد. یعنی برای سه فیلد، باید شش متد بنویسیم. آیا این روش کمی‌پردردسر نیست ؟

برای دسترسی به مقدار فیلدها که private شده‌اند باید متدهای نوشته شده را در برنامهٔ اصلی فراخوانی کنیم. بدنهٔ متد EH رویداد کلیک دکمه showButton به صورت زیر تغییر خواهد کرد :

```
string s = "speed: " + basicSpaceShip.GetSpeed ( )+
           "\n fuel: " +basicSpaceShip.GetFuel( )+
           "\n shield: " +basicSpaceShip.GetShield ( );
MessageBox.Show(s, "مقدار خصوصیات سفینه");
```

روش سوم ــ استفاده از ویژگی (Property) : با توجه به مشکلاتی که در روش‌های اول و دوم داشتیم راه حلی که در زبان C# پیش بینی شده است استفاده از قابلیت ویژگی (Property) است. در این روش متدی نوشته می‌شود که شبیه یک فیلد مورد استفاده برنامه نویس قرار می‌گیرد. روش نوشتن نام ویژگی‌ها مانند نام متدها، روش پاسکال است. مثلاً تعریف یک ویژگی برای دسترسی کنترل شده به فیلد سوخت چنین است :

```
public class SpaceShip
{
    private int fuel;
    private int shield;          فیلدها
    private int speed;

    public int Fuel
    {                            ویژگی
        get { return fuel; }
        set { fuel = value;}
    }
}
```

به روش تعریف ویژگی Fuel دقت کنید. باید قبول کنیم که در تعریف یک ویژگی، ترکیبی از روش تعریف فیلد و روش تعریف متد با هم استفاده شده است. معمولاً نام ویژگی‌ها را مانند نام فیلدها انتخاب می‌کنند با این تفاوت که روش نوشتن نام ویژگی از روش پاسکال است. کلمات رزروشده

get ، set، value و return در تعریف یک ویژگی استفاده می‌شود این چهار کلمه کلیدی در جدول زیر توضیح داده شده است :

جدول ۹ــ۴ــ کلمات کلیدی مورد استفاده در تعریف ویژگی

کلمه کلیدی	شرح کار
get	بخشی از تعریف ویژگی را مشخص می‌کند که برای اطلاع از مقدار یک فیلد به‌کار می‌رود.
set	بخشی از تعریف ویژگی را مشخص می‌کند که برای انتساب یک مقدار به یک فیلد به‌کار می‌رود. در این بخش می‌توانیم با استفاده از دستور if مقداری که قرار است در فیلد قرار گیرد را کنترل و بررسی کنیم.
value	مقداری است که در برنامه معین شده و قرار است در فیلد قرار گیرد.
return	دقیقاً عملکردی مانند مقدار برگشتی متدها به برنامه اصلی را دارد.

در تعریف ویژگی Fuel برای سادگی کار، کنترلی بر روی value صورت نگرفته است اما در تعریف زیر برای مقادیر ویژگی‌های دیگر چنین کاری انجام شده است :

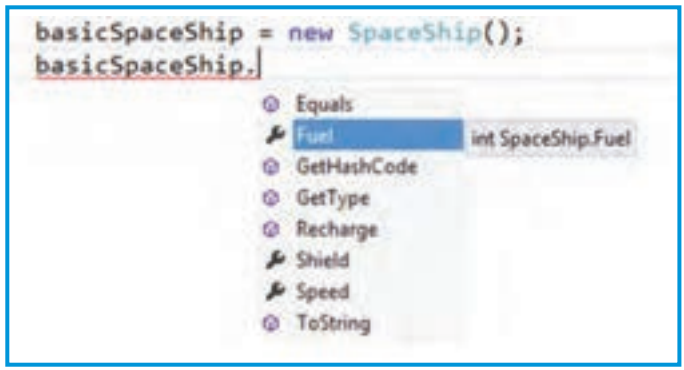
```
public class SpaceShip
{
    private int fuel;
    private int shield;
    private int speed;

    public void Recharge ( )
    {
        fuel = 100;
        shield = 100;
        speed = 100;
    }
```

```
public int Fuel
{
        get { return fuel; }
        set { fuel = value;}
}
public int Shield
{
        get { return shield; }
        set { if ((value< =100) && (value>=0))
                shield=value;
            }
}
public int Speed
{
get { return speed; }
        set { if ((value< =100) && (value>=0))
                speed=value;
            }
}
}
```

آیا می‌توانید تشخیص دهید هر کادر رنگی معرف چه عضوی از یک کلاس است؟ پاسخ شما باید یکی از کلمات فیلد، متد و ویژگی باشد.

به‌تدریج که جلو می‌رویم، تعریف کلاس سفینه کامل می‌شود و شما با مفاهیم پایه‌ای در تعریف یک کلاس و زبان شیء گرایی آشنا می‌شوید.

اکنون نوبت نمایش ویژگی‌های سفینه است. از ویژگی‌ها مانند یک متغیر برای اطلاع از وضعیت سفینه و همچنین تغییر مقدار فیلدها استفاده می‌کنیم.



شکل ۱۶ــ۴

در لیست هوشمند (IntelliSense)، ویژگی‌ها با علامت آچار مشخص می‌شوند. نمونه ای از برنامه اصلی چنین خواهد بود :

```
public partial class Form1 : Form
{
    SpaceShip basicSpaceShip;
    public Form1( )
    {
        InitializeComponent ( );
        basicSpaceShip = new SpaceShip ( );
    }
    private void showButton_click (object sender, EventArgs e)
    {
        string s = "speed: " +basicSpaceShip.Speed+
                   "\n fuel: " +basicSpaceShip.Fuel+
                    "\n shield: " +basicSpaceShip.Shield;
        MessageBox.Show(s, "مقدار خصوصیات سفینه");
    }
}
```

مثال۳ــ۴ : می‌خواهیم در برنامهٔ اصلی دو شیء از نوع سفینه ایجاد کرده و اندازهٔ سرعت یکی ۱۰۰ و دیگری را برابر ۵۰ تنظیم کنیم.

الگوریتم یا روش انجام کار : برای ایجاد دو سفینه کافی است دو متغیر مرجع مختلف تعریف کرده و با استفاده از عملگر new، عمل ایجاد شیء را انجام دهیم . سپس ویژگی Speed هر یک را مانند یک متغیر معمولی با دستور انتساب مقدار دهی کنیم.

```
public partial class Form1 : Form
{
    SpaceShip basicSpaceShip1, basicSpaceship2;
    public Form1( )
    {
        InitializeComponent ( );
        basicSpaceShip1 = new spaceShip ( );
        basicSpaceShip2 = new spaceShip ( );
        basicSpaceShip1 . Speed = 100;
        basicSpaceShip2 . Speed = 50;
    }
```

سؤال: در برنامهٔ بالا مقدار سوخت و سپر دفاعی هر یک از سفینه‌ها چقدر است؟

برای پاسخ به سؤال کافی‌است برنامه را اجرا کرده و روی دکمه showButton کلیک کنیم تا مقدار تمام ویژگی‌های سفینه ۱ را در پنجره پیام نمایش دهد.

```
private void showButton_click (object sender, EventArgs e)
{
    string s = "speed: " + basicSpaceShip1 . Speed+
               "\n fuel: " + basicSpaceShip1 . Fuel+
               "\n shield: " + basicSpaceShip1 . Shield;
    MessageBox . Show(s, "مقدار خصوصیات سفینه");
}
```

شکل ۱۷ــ۴ــ متد Eh رویداد کلیک دکمه showButton

شکل ۱۸ــ۴ــ خروجی مثال ۳ــ۴

سؤال: چرا مقدار فیلدهای سوخت و سپر دفاعی صفر چاپ شد؟ چگونه آنها مقداردهی شدند؟ همان طور که به یاد دارید، چنانچه متغیری تعریف کنیم و آن را مقداردهی نکنیم، مترجم خطا می‌دهد. اما در تعریف کلاس سفینه، مقدار فیلدها را مقداردهی نکردیم اما مترجم ایرادی نگرفته است.

پاسخ این سؤالات چنین است که به هنگام ایجاد یک شیء، یک متد که به نام متد سازنده[۱] معروف است به طور خودکار و پیش فرض اجرا می‌گردد و فیلدهای شیء را مقداردهی و معین می‌کند. اگر ما متدی را به عنوان متد سازنده معرفی نکنیم، هنگام ایجاد شیء اگر فیلدی مقداردهی اولیه نشده باشد مقدار پیش فرض به آن داده می‌شود. مقدار پیش فرض برای فیلدهای نوع عددی، عدد صفر و برای فیلدهای رشته ای مقدار تهی یا null است.

۹ــ۴ــ متد سازنده

یکی از راه‌های مقداردهی اولیه به فیلدها و در واقع راه اصلی آن استفاده از متد سازنده است. متد سازنده، متدی است که در هنگام ایجاد شیء فراخوانی شده و عملیات آن انجام می‌شود. در واقع هرگاه بخواهیم در هنگام ایجاد یک شیء، عملیاتی انجام شود از متد سازنده استفاده می‌کنیم. این عملیات می‌تواند مقداردهی اولیه فیلدها یا هر دستور دیگری باشد .

در تعریف یک کلاس می‌توانیم متد سازنده‌ای به دلخواه و طبق نیاز برنامه تعریف کنیم که در این صورت این متد با ایجاد شیء به طور خودکار اجرا می‌گردد. نام متد سازنده هم نام با نام کلاس است و نوعی به‌عنوان خروجی نباید برای آن معین شود. متد سازنده می‌تواند با پارامترو یا بدون پارامتر باشد.

۱ــ Constructor

علاوه بر آن می‌توان چندین متد سازنده، داشت یعنی چندین متد هم نام با نام کلاس داشته باشیم که تفاوت آنها در پارامترهای ورودی متد است.

مثلاً فرض کنید می‌خواهیم با ایجاد یک سفینه، مقادیر همه فیلدها برابر ۵۰ شود. در این صورت خواهیم داشت :

```
public class SpaceShip
{
    private int fuel;
    private int shield;
    private int speed;
    public Spaceship ( )                    متد سازنده
    {
        fuel = 50;
        shield = 50;
        speed = 50;
    }
    public void Recharge ( )
    {
        fuel = 100;
        shield = 100;
        speed = 100;
    }
        ادامه تعریف کلاس
```

اگر در برنامهٔ اصلی یک سفینه ایجاد کنیم متد سازنده به طور خودکار اجرا شده و مقدار فیلدها برابر ۵۰ می‌شود. بنابراین با اجرای همان برنامه قبل پنجره پیام مانند شکل ۱۹ــ۴ خواهد بود.

شکل ۱۹ــ۴

مثال ۴ــ۴ : می‌خواهیم در هنگام ایجاد یک سفینه، اگر مایل بودیم اندازهٔ سوخت، سپر دفاعی و سرعت سفینه با اعداد دلخواه تعیین شود.

الگوریتم یا روش انجام کار : چون می‌خواهیم در هنگام ایجاد شیء، مقداردهی صورت گیرد لذا باید از متد سازنده بدین منظور استفاده کنیم. بنابراین کافی است که متد سازندهٔ دیگری به تعریف کلاس مثال قبل اضافه نماییم. متد سازنده جدید باید دارای سه پارامتر ورودی باشد که هر یک از فیلدهای سفینه با پارامتر مربوطه مقداردهی شود :

```
public  Spaceship (int f , int sh , in sp)
{
    Fuel = f;
    Shield = sh;
    Speed = sp;
}
```

برنامهٔ اصلی چنین خواهد بود :

```
public partial class Form1 : Form
{
    SpaceShip basicSpaceShip1, basicSpaceship2;
    public Form1( )
    {
        InitializeComponent ( );
        basicSpaceShip1 = new spaceShip (100,80,90);← سازنده با پارامتر ورودی
        basicSpaceShip2 = new spaceShip ( );  ← سازنده بدون پارامتر ورودی
    }
    private void showButton_click (object sender, EventArgs e)
    {
        string s = "safineh1\t \t safineh2 \n" +
                   "speed: " + basicSpaceship1. Speed +
                   "\t \t speed: " + basicSpaceShip2.Speed +
                   "\n fuel: " + basicSpaceShip1.Fuel +
                   "\t  \t fuel: " + basicSpaceship2.Fuel +
                   "\n shield: " + basicSpaceShip1.Shield +
                   "\t  \t  shield: " + basicSpaceShip2. Shield;
        MessageBox.Show(s,  "مقدار خصوصیات سفینه");
    }
}
```

در این برنامه basicSpaceShip1 براساس سازنده دوم که دارای ۳ ورودی برای مقادیر speed, shield و fuel است ساخته شده ولی basicSpaceShip۲ براساس سازندهٔ اوّل ساخته شده است که ورودی ندارد و مقادیر تمام ویژگی‌ها را ۵۰ قرار می‌دهد (سازنده بدون ورودی را سازنده پیش فرض[۱] می‌گویند) و نتیجه کلیک دکمه showButton نمایش پنجره پیام مانند شکل ۲۰ــ۴ است. (\t معادل کلید tab عمل کرده و چند کاراکتر فاصله در پیام قرار می‌دهد.)

شکل ۲۰ــ۴ــ خروی مثال ۴ــ۴

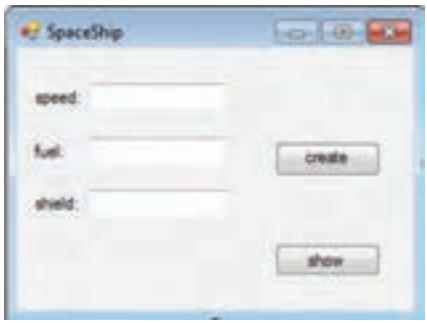
به قطعه کد زیر دقت کنید.

```
basicSpaceShip1 = new SpaceShip(100,80,90);
basicSpaceShip2 = new SpaceShip.SpaceShip(int f, int s, int sh) (+ 1 overload(s))
```

سؤال: چرا در کنار توضیح ظاهر شده به وسیله IntelliSense پیام (+۱ Overload) دیده می‌شود؟

بهبود و توسعه برنامه

۱ــ فرمی مانند شکل ۲۱ــ۴ ایجادکرده و از سه کادر متنی برای مقداردهی ویژگی‌های سفینه و دکمه CreateButton برای ایجاد یک سفینه استفاده کنید. کاربر تنها اجازه ایجاد یک سفینه را داشته باشد و امکان ورود کاراکترهای غیر عددی در کادرهای متنی نباشد.



شکل ۲۱ــ۴ــ فرم توسعه و بهبود مثال ۴ــ۴

۱ــ Default constructor

۲ــ سه سفینه بسازید. برای نمایش اطلاعات هر یک از آنها تصویری روی فرم قرار دهید. در صورت کلیک هر یک از تصاویر اطلاعات سفینه نظیر آن در برچسب نمایش داده شود.

شکل ۲۲ــ۴ــ فرم نمایش اطلاعات سه سفینه

نکته

بهترین الگو برای شیء گرایی با مفهوم خاص آن در برنامه نویسی و ساز و کار مستقل اشیاء و تعامل آنها با یکدیگر همین جهان پیرامون ماست. خوب به هر شیء و ویژگی‌ها و رفتارهای آن بنگرید ، دربارهٔ آن فکر کنید و سعی کنید آن را به صورت یک کلاس پیاده‌سازی کنید. تمرین زیاد باعث می شود که در هنگام برنامه، آسانتر اشیای موجود در سیستم و ویژگی‌هایشان را تشخیص بدهید و کلاس مربوط به آنها را بنویسید.

شکل ۲۳ــ۴

کار در کارگاه۵: حساب پس‌انداز شخصی

مثال ۵ــ۴ : می‌خواهیم یک حساب شخصی داشته باشیم که بتوانیم به آن مبلغ دلخواه خود را واریز کنیم یا از آن برداشت کنیم و در هر لحظه نیز مبلغ پس انداز ما مشخص باشد.

شکل ۲۴ــ۴ــ فرم حساب پس‌انداز شخصی

الگوریتم یا روش انجام کار : برای داشتن این حساب پس انداز، ابتدا یک کلاس حساب (account) می‌سازیم که دارای عملیات افزایش و کاهش باشد و همچنین مقدار پس انداز را نیز بتوانیم از آن بگیریم. با زدن دکمه واریز، مبلغ وارد شده به حساب اضافه می‌شود و با زدن دکمه برداشت، مبلغ وارد شده از حساب کم می‌شود. باید توجه کرد که تغییر در حساب پس انداز تنها از طریق این دو عملیات، انجام می‌شود.

۱ــ ایجاد پروژه : ابتدا یک پروژه ایجاد می‌کنیم و فرم ایجاد شده را به‌صورت زیر مقداردهی می‌نماییم.

جدول ۱۰ــ۴ــ ویژگی‌های فرم

Form		
ویژگی	مقدار	
Name	accountForm	
Text	حساب پس‌انداز من	
Size	width	۴۹۰
	height	۲۶۰

۲ــ اضافه کردن کنترل‌ها : کنترل‌های فرم را نیز به صورت زیر اضافه می‌کنیم.

جدول ۱۱ــ۴ــ ویژگی‌های GroupBoxها

GroupBox		
ویژگی	**مقدار**	**مقدار**
Name	groupBox۱	groupBox۲
Text	مبلغ پس انداز شده	واریز ــ برداشت

کنترل برچسب را در groupBox1 اضافه می‌کنیم.

جدول ۱۲ــ۴ــ ویژگی‌های برچسب

Label		
ویژگی	**مقدار**	
Name	savingLbl	
Text		
Background	white	
AutoSize	False	
RightToLeft	Yes	
Font	Size	۱۸
Size	width	۲۰۰
	height	۶۲

یک برچسب، کادر متن و دکمه در group Box2 قرار دهید و ویژگی‌های آنها را مطابق جداول زیر تنظیم کنید.

جدول ۱۳ــ۴ــ ویژگی‌های برچسب

Label	
ویژگی	**مقدار**
Name	label۱
Text	مبلغ :
AutoSize	True
RightToLeft	Yes

جدول ۱۴ــ۴ــ ویژگی‌های کادر متن

TextBox	
ویژگی	**مقدار**
Name	amountTb
Text	
RightToLeft	Yes
width	۱۹۱

جدول ۱۵ــ۴ــ ویژگی‌های دکمه

Button		
ویژگی	**مقدار**	**مقدار**
Name	addBtn	reduceBtn
Text	واریز	برداشت

۳ــ تعریف کلاس و شیء : پس از طراحی واسط کاربری، به سراغ کدها می‌رویم. در ابتدا درون کلاس فرم، کلاس account را برای حساب پس‌انداز ایجاد می‌کنیم و پس از تعریف کلاس، یک شیء از آن می‌سازیم.

```
public partial class accountForm : Form
{
  class Account
  {
        private uint saving:
        public uint Saving { get {return saving; } }
        public void Add(uint amount) { saving + = amount; }
        piblic void Reduce(uint amount) { if (amount <=
        saving) saving − = amount; }
        public Account (uint amount) { saving = amount; }
  }
Accont myAccount = new Account (0);
```

تعریف کلاس

ایجاد شیء

موارد قابل توجه کلاس Account :

✓ فیلد saving برای نگهداری پس‌انداز تعریف شده است.

✓نوع uint فقط اعداد صحیح بزرگ‌تر یا مساوی صفر را می‌پذیرد. دلیل انتخاب این نوع برای مقدار پس‌انداز این است که پس‌انداز منفی قابل قبول نیست.

✓ ویژگی Saving فقط متد get دارد بنابراین فقط خواندنی است.

✓ متد Add برای اضافه کردن مقداری به پس انداز ایجاد شده است.

✓ متد Reduce برای برداشت از حساب تعریف شده است.

✓متد سازنده کلاس، یک ورودی دارد تا مبلغ اولیه افتتاح حساب مشخص شود.

همان‌طور که گفته شد برای مقدار حساب پس انداز و مقادیر واریزی یا برداشتی از uint استفاده شده است. اما این نوع، مقدار زیادی را در خود جای می‌دهد که برای حساب، استفاده‌ای نمی‌شود. آیا نوع کوچک‌تری را می‌شناسید که بتوان جایگزین آن کرد؟

در مورد ایجاد حساب myAccount باید گفت که به‌دلیل نیاز به نگهداری آن در طول برنامه و

استفاده از آن در رویدادهای مختلف، در خود کلاس فرم accountForm و به عنوان یک فیلد سراسری آن تعریف شده است. ضمناً از نوع ایجاد آن مشخص است که حساب دارای مقدار اولیه ۰ است.

پس از فراخوانی متد مقداردهی اولیه فرم (InitializeComponent()) در متد سازنده فرم accountForm، برچسب پس‌انداز (savingLbl) را مقداردهی می‌کنیم تا در همان ابتدای ایجاد فرم، مقدار پس انداز مشخص باشد.

```
public accountForm()
{
    InitializeComponent();
    savingLbl.Text = myAccount.Saving.ToString() + "ریال";
}
```

نمایش مقدار پس‌انداز در برچسب

متد رویدادهای کلیک دو دکمه واریز و برداشت را به‌صورت زیر اضافه می‌نماییم.

```
private void addBtn_Click(object sender, EventArgs e)
{
  myAccount.Add(uint.Parse(amountTb.Text));
  savingLbl.Text = myAccount.Saving.ToString( ) + "ریال";
}

private void reduceBtn_Click(object sender, EventArgs e)
{
   myAccount.Reduce(uint.Parse(amountTb.Text));
  savingLbl.Text = myAccount.Saving.ToString() + "ریال";
}
```

در متد رویداد کلیک دکمه واریز، متد Add از شیء myAccount برای اضافه کردن مقدار وارد شده در کادر متنی amountTb اجرا می‌شود. اما محتوای کادر متنی با متد uint.Parse تبدیل به نوع uint می‌شود. در خط دوم مقدار پس‌انداز به وسیلهٔ برچسب نمایش داده می‌شود.

در متد رویداد کلیک دکمه برداشت، دستورات مشابه دکمه واریز است با این تفاوت که به جای متد Add از متد Reduce استفاده می‌شود.

توسعه و بهبود برنامه

در کادر متنی مبلغ ، هر مقداری از جمله حروف الفبایی و اعداد منفی را می‌شود وارد کرد که باعث خطا در برنامه می‌شوند. برنامه را طوری تغییر دهید که در صورت ورود مقادیر غیرمجاز، پیام مناسبی از طرف برنامه داده شود.

خودآزمایی فصل چهارم

الف) درستی یا نادرستی عبارات زیر را تعیین کنید :

۱ــ ویژگی، بیانگر عملیات یا رفتار شیء در کلاس است.

۲ــ هر شیء که ساخته می‌شود به میزان مورد نیاز، فضایی از Ram به آن اختصاص داده می‌شود.

۳ــ متدها به مقدار لازم، ورودی دارند.

۴ــ متدها بیش از یک مقدار برگشتی یا خروجی دارند.

۵ــ اگر نوع خروجی متدی Void باشد، متد مقداری را برنمی‌گرداند.

۶ــ برای انتساب مقداری به یک فیلد کلاس، از کلمه کلیدی get استفاده می‌شود.

۷ــ در تعریف کلاس، اگر مقدار فیلدها مقداردهی نشود مترجم خطا می‌گیرد.

ب) جاهای خالی را با عبارت مناسب پر کنید :

۸ــ در متد، برای برگرداندن مقدار خروجی از کلمه کلیدی ..... استفاده می‌شود.

۹ــ در زبان سی‌شارپ برای ایجاد یک شیء از یک کلاس، از عملگر ..... استفاده می‌شود.

۱۰ــ برای اطلاع از مقدار یک ویژگی در کلاس، در هنگام تعریف آن، کلمه کلیدی ..... به‌کار برده می‌شود.

۱۱ــ هنگام ایجاد شیء مقدار پیش‌فرض برای فیلدهای رشته‌ای که مقداردهی اولیه نشده ..... است.

ج) به سؤالات زیر پاسخ دهید :

۱۲ــ تفاوت کلاس و شیء را با مثال بنویسید.

۱۳ــ تفاوت دو توصیف‌کننده public و private را بنویسید.

۱۴ــ متد سازنده چه متدی است؟

تمرینات برنامه‌نویسی فصل چهارم

۱ــ برنامه‌ای برای نمایش اطلاعات دانش‌آموز بنویسید. برای نوشتن این برنامه کارهای زیر را انجام بدهید :

- کلاسی برای دانش‌آموز شامل ویژگی‌های فردی (کد دانش‌آموز ــ نام ــ نام‌خانوادگی ــ نام پدر) و نمرات ۳ درس دانش‌آموز تعریف کنید.
- سه متد سازنده، تعریف کنید که متد اول مقدار یک نمره، متد دوم مقدار دو نمره و متد سوم مقدار سه نمره را تعیین کند.
- متدی در کلاس قرار دهید که مقدار برگشتی آن، معدل این سه نمره باشد.
- از برچسب برای نمایش ویژگی‌های فردی، نمره‌ها و معدل و اطلاعات دانش‌آموز برروی فرم استفاده کنید. برای ایجاد شیء مربوط به هر دانش‌آموز، از یکی از متدهای سازنده استفاده نمایید.

۲ــ مثال ساعت در متن درس را به‌صورت زیر تغییر دهید.

- یک شیء Timer به فرم اضافه کرده، ویژگی interval آن را برابر با ۱۰۰۰ قرار دهید تا در هر ثانیه رویداد آن اجرا شود. این timer در ابتدا فعال است.
- متدی به نام AddSecond به کلاس ساعت اضافه نمایید که زمان سپری شده را محاسبه نماید.

راهنمایی : این متد ثانیه‌ها را می‌شمارد و در هر فراخوانی یک ثانیه به مقدار زمان اضافه می‌کند. با رسیدن ثانیه به ۶۰ باید مقدار دقیقه یک واحد اضافه شود و همین‌طور مقدار ساعت نیز در

هنگام رسیدن دقیقه به ۶۰ یک واحد افزوده شود.

o در رویداد Tick تایمر، متد AddSecond شیء ساعت را فراخوانی کنید.

تحقیق: می‌توان timer را درون کلاس ساعت ایجاد نمود. مزیت و چگونگی این کار را بررسی کنید.

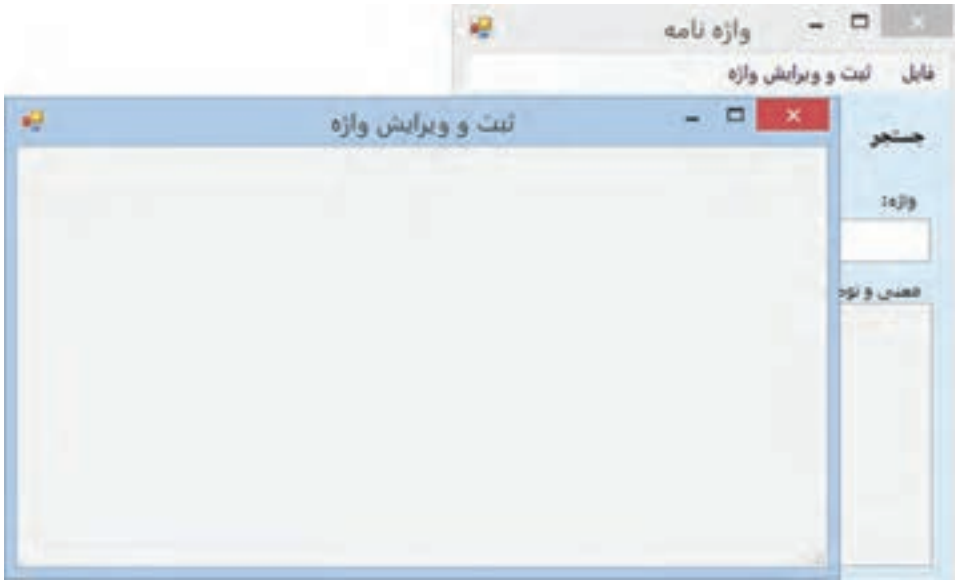
فعالیت

تکمیل پروژه

در این فصل با تعریف کلاس فرم، فضایی فراهم می‌کنیم تا در آن امکان افزودن و ویرایش واژه‌ها صورت گیرد.

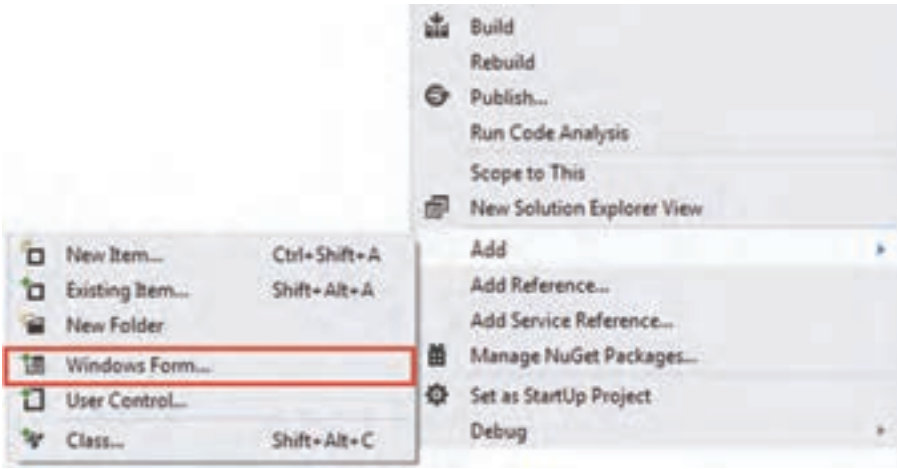
مرحله چهارم:

در این مرحله می‌خواهیم با انتخاب گزینه «ذخیره جستجو» فرمی باز شود تا در مراحل بعدی بتوان از این فرم برای اضافه کردن واژه‌های جدید یا ویرایش واژه‌های موجود از آن کمک گرفت. همچنین می‌خواهیم قسمتی از کد ذخیره جستجو هم در این مرحله نوشته شود.



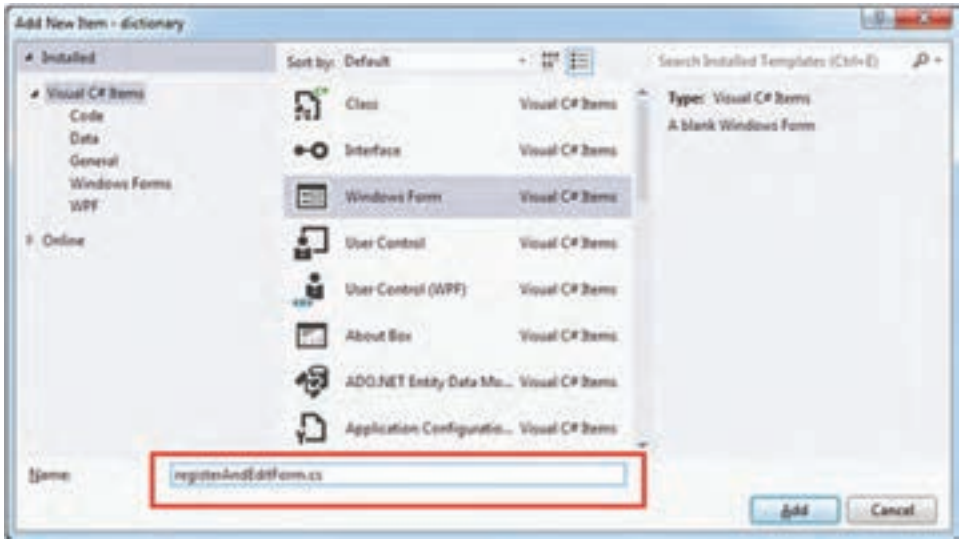
مراحل کار : برای ایجاد فرم جدید باید ابتدا کلاس فرم جدید را تعریف کرد. این کار با استفاده از امکانات ویژوال استودیو بسیار آسان است.

۱ــ روی نام پروژه کلیک راست نمایید تا منوی آن نمایش داده شود. گزینه Windows Form را از منوی Add کلیک نمایید.



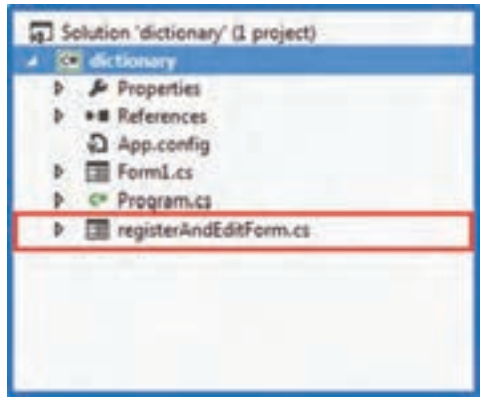
افزودن فرم جدید

۲ــبا کلیک این گزینه، کادر افزودن بخش جدید باز می‌شود و به‌صورت خودکار گزینه Windows Form انتخاب شده است.



ذخیره فرم

با وارد کردن نام فرم جدید به صورت registerAndEditForm.cs و زدن دکمه Add فرم جدید ایجاد خواهد شد.



ویژگی‌های فرم جدید را به صورت زیر تغییر دهید :

شیء Form		
ویژگی		مقدار
Name		registerAndEditForm
Text		ثبت و ویرایش واژه
Size	Width	519

۳ــ متد EH رویداد کلیک گزینه "ثبت و ویرایش واژه" در فرم اصلی را به صورت زیر تغییر دهید.

```
private void registerAndEditMenuItem_Click(object sender,
EventArgs e)
{
  registerAndEditForm regAndEdit = new registerAndEditForm();
  regAndEdit.ShowDialog();
}
```

در خط اول کد داخل متد، یک نمونه (شیء) از کلاس registerAndEditForm ساخته می‌شود و در متغیر regAndEdit قرار می‌گیرد.

با استفاده از متد ShowDialog در خط دوم، فرم ساخته شده (regAndEdit) نمایش داده می‌شود. نحوه نمایش این فرم به صورتی است که مانند یک کادر محاوره ای نمایش داده می‌شود و فرم اصلی قابل دسترسی نخواهد بود

سؤال: بررسی کنید اگر به جای متد ShowDialog از متد Show استفاده کنید چه تفاوتی در نمایش فرم دارد؟

۴ــ برای گزینه «ذخیره جستجو» در منو باید کادر ذخیره فایل باز شود تا محل و نام فایل ذخیره از کاربر پرسیده شود. در فصل بعد خواهید آموخت چگونه فایل را ذخیره کنید. فایلی که توسط این نرم افزار ذخیره می‌شود حاوی اطلاعات واژه مورد جستجو است.

متد EH کلیک گزینه «ذخیره جستجو» را که قبلاً ساخته‌اید به‌صورت زیر تغییر دهید.

```
private void saveMenuItem_Click(object sender, EventArgs e)
{
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.Filter = "text file|*.txt";
}
```

در خط اول این متد، یک شیء از کلاس SaveFileDialog ساخته می‌شود و آدرس آن در متغیر sfd قرار داده می‌شود. در خط دوم ویژگی Filter طوری مقداردهی می‌شود که ذخیره فایل با پسوند txt باشد.

واژگان و اصطلاحات انگلیسی فصل چهارم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Account	
۲	Add	
۳	Armor	
۴	Battle	
۵	Cargo Spaceship	
۶	Class	
۷	Constructor	
۸	Fire Rate	
۹	Get	
۱۰	Initialize	
۱۱	Null	
۱۲	Object	
۱۳	Property	
۱۴	Private	
۱۵	Public	
۱۶	Reduce	
۱۷	Reset	
۱۸	Return	
۱۹	Set	
۲۰	Shield	
۲۱	Speed	
۲۲	Storage	
۲۳	Value	
۲۴	Void	

فصل پنجم

فایل

در بیشتر برنامه‌هایی که تاکنون نوشتیم با داده‌هایی سروکار داشتیم که به طور موقتی[1] در داخل متغیّرها و یا آرایه قرار داشتند که معمولاً پس از خروج از برنامه، داده‌ها مورد استفاده قرار نمی‌گرفتند. در این فصل می‌خواهیم داده‌ها را در حافظه‌های جانبی ذخیره کنیم تا بعد از خروج از برنامه نیز آنها را در اختیار داشته باشیم.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ ضرورت و اهمیت ایجاد فایل را بیان کند.

۲ــ یک فایل متنی ایجاد نماید و اطلاعات را در آن ذخیره کند.

۳ــ فایل متنی را خوانده و محتویات آن را روی فرم نمایش دهد.

۴ــ از متدهای مختلف برای عملیات بر روی فایل‌ها استفاده نماید.

۱– Temporary

۱ــ۵ــ فایل چیست

دنباله‌ای[۱] از بایت‌ها را که در روی حافظه‌های جانبی تحت یک نام نگهداری می‌شود، فایل می‌نامند. از نظر محتوا، انواع مختلفی از فایل‌ها وجود دارند. فایلی که محتوای آن، کاراکترهای چاپ شدنی است، **فایل متنی**[۲] نامیده می‌شود. در انتهای فایل‌های متنی، یک بایت به عنوان پایان فایل نشانه‌گذاری می‌شود که به آن کاراکتر پایان فایل[۳] می‌گویند. محتوای فایل‌های متنی با یک ویرایشگر ساده مانند Notepad و یا یک واژه‌پرداز مانند Word قابل مشاهده می‌باشد.

اگر داده‌های موجود در متغیّرهای یک برنامه که شامل اعداد و رشته‌ها است را به همان شکل که در حافظه قرار دارند، در یک فایل ذخیره نماییم، آنگاه فایل ایجاد شده را **فایل دودویی**[۴] می‌نامند. محتوای فایل‌های دودویی را نمی‌توان با برنامه‌های ویرایشگر یا واژه‌پرداز نظیر Notepad یا Word به درستی مشاهده کرد.

از نظر ترتیب ذخیره‌سازی اطلاعات و یا نحوه دسترسی به اطلاعات درون یک فایل، انواع مختلفی از فایل‌ها وجود دارند. فایل با دسترسی ترتیبی[۵] و فایل با دسترسی مستقیم[۶]. فایل‌ها درون حافظه‌های جانبی ذخیره می‌شوند تا برای دسترسی‌های بعدی مورد استفاده قرار بگیرند.

در کتابخانه NET . در فضای نامی System.IO کلاس FileStream برای انجام عملیات بر روی فایل‌ها تعریف شده است. همچنین کلاس‌های File و Directory و Path که شامل تعدادی متد استاتیک است به ترتیب برای انجام عملیات مختلف بر روی فایل‌ها، فولدرها و همچنین مسیر دسترسی به فایل‌ها و فولدرها به کار می‌رود. برای آشنایی بیشتر با متدهای این کلاس به پیوست ۵ مراجعه کنید. در جدول ۱ــ۵ چند متد استاتیک پر استفاده از کلاس File را مشاهده می‌کنید.

۱ــ Stream

۲ــ Text File

۳ــ End of file

۴ــ Binary file

۵ ــ Sequential access

۶ــ Direct access

جدول ۱ــ۵ــ برخی متدهای کلاس file

ردیف	نام متد	شرح
۱	Exists()	اگر فایل وجود داشته باشد مقدار خروجی متد برابر true است در غیر این صورت مقدار false برمی‌گرداند.
۲	WriteAllText()	می‌توانید یک فایل متنی جدید ایجاد کرده و متنی در آن بنویسید و سپس فایل را ببندید. اگر فایل وجود داشته باشد روی آن بازنویسی می‌کند.
۳	AppendAllText()	می‌توانید یک فایل متنی را باز کرده و متنی را به انتهای آن اضافه کرده و سپس فایل را ببندید.
۴	ReadAllText()	یک فایل متنی موجود را باز کرده و محتوای آن را خوانده و به صورت یک رشته بر می‌گرداند. در پایان فایل را می‌بندد.

کار در کارگاه۱: ایجاد یک فایل متنی

مثال۱ــ۵ : می‌خواهیم برنامه‌ای بنویسیم که اطلاعات وارد شده در فرم شکل۱ــ۵ را در داخل فایل متنی ذخیره نماید.

شکل ۱ــ۵ــ فرم دریافت نام کاربری و کلمه عبور

الگوریتم یا روش انجام کار : ایجاد فرمی‌برای ورود اطلاعات نام کاربری و کلمه عبور به سادگی انجام می‌شود. هنگامی که کاربر بر روی دکمه "ذخیره" کلیک می کند، باید عملیات زیر انجام شود :

۱ــ کنترل صحت اطلاعات صورت گیرد. آیا نام وارد شده به عنوان نام کاربری مجاز است. آیا کلمه عبور وارد شده با تکرار کلمه عبور یکسان است؟

۲ــ بعد از کنترل صحت اطلاعات وارد شده، باید یک فایل متنی ساخته شود و اطلاعات نام کاربری و کلمه عبور در آن ذخیره شود.

۳ــ پیامی مبنی بر ذخیره سازی موفق اعلام شود.

اکنون برای انجام پروژه، عملیات زیر را انجام می‌دهیم :

۱ــ ایجاد یک پروژه ویندوزی: وارد برنامه VS شوید و یک پروژه جدید از نوع WFA با نام TextFileDemo و در مسیر مشخصی بسازید. اگر اکنون داخل VS هستید از طریق منوی File، انتخاب گزینه Close Solution، پروژه فعلی را ببندید و سپس یک پروژه جدید مطابق با آنچه که گفته شد، بسازید.

۲ــ تعیین ویژگی‌های فرم: ویژگی‌های فرم را به صورت جدول ۲ــ۵ تنظیم کنید.

جدول ۲ــ۵ــ مقادیر ویژگی‌های شیء فرم

Form		
ویژگی		مقدار
Name		Form1
Text		ثبت نام کاربر جدید
Size	Width	۳۰۰
	Height	۳۰۰

۳ــ اضافه کردن سه برچسب بر روی فرم: سه کنترل برچسب بر روی فرم برای عناوین "نام کاربری" ، "کلمه عبور" و "تکرار کلمه عبور" مانند شکل ۱ــ۵ قرار دهید و ویژگی‌های آن را تنظیم کنید.

۴ــ اضافه کردن سه کادر متنی برای دریافت اطلاعات کاربر: سه کادر متنی برای

دریافت اطلاعات کاربر شامل نام کاربری ، کلمه عبور و تکرار کلمه عبور مانند شکل ۱ــ۵ به فرم اضافه نمایید. ویژگی‌های آنها را تنظیم کنید. توجه داشته باشید که نام آنها را مانند جدول زیر انتخاب کنید.

جدول ۳ــ۵ــ ویژگی کادرهای متنی

TextBox		کادرهای متنی	
ویژگی	مقدار	مقدار	
Name	userNameText	passwordText	verifiedText
Password		*	*

۵ــ اضافه کردن دو دکمه : دو دکمه نیز به فرم اضافه کنید. خصوصیات آنها را طوری تنظیم کنید که مانند شکل ۱ــ۵ قرار گیرند. نام آنها را مطابق با جدول ۴ــ۵ انتخاب کنید :

جدول ۴ــ۵ــ ویژگی دکمه‌ها

دکمه Button		
ویژگی	مقدار	مقدار
Name	saveButton	clearButton
Text	ذخیره	پاک کردن

۶ــ اضافه کردن یک کادر تیک :
یک کادر تیک به فرم اضافه کنید تا برای نمایش کلمه عبور استفاده شود. ویژگی‌های آن را مطابق با جدول ۵ــ۵ تنظیم کنید.

جدول ۵ ــ ۵ــ ویژگی کادر تیک

CheckBox	
ویژگی	مقدار
Name	checkBox1
Text	نمایش کلمه عبور
RightToLeft	Yes

۷ــ نوشتن متدهای EH

الف ــ متد EH کادرتیک را به صورت زیر بنویسید :

```
Private void checkBox۱ (underline)_ CheckedChanged (object
sender,EventArgs e)
{
    if (passwordText.PasswordChar == '*' )
    {
        passwordText.PasswordChar = '\u0000 ' ;
        verifiedText.PasswordChar = (char)0;
    }
    else
    {
        passwordText.PasswordChar = '*';
        verifiedText.PasswordChar = '*';
    }
}
```

نکته

اگر ویژگی PasswordChar مقداردهی شود (با کاراکتری مانند*) به جای نمایش محتوای کادر متن، تمام کاراکترهای محتوا با کاراکتر تعیین شده در ویژگی PasswordChar جایگزین می‌شود. برای برگشت کادرمتن به حالت عادی باید این ویژگی، با کاراکتری با کد اسکی صفر[۱] مقداردهی شود.

در کد متد از دو روش برای مقداردهی ویژگی PasswordChar استفاده شده است. اولی با استفاده از کد اسکی ۴ رقمی کاراکتر (\u۰۰۰۰) و دومی استفاده از تبدیل نوع که عدد صفر را به نوع کاراکتر تبدیل کرده است.

۱ــ null

ب ــ نوشتن متد EH برای دکمه‌ها

ــ متد EH دکمه، "پاک کردن فرم" : وظیفه متد EH دکمه "پاک کردن"، حذف اطلاعات درون کادرهای متنی است.

سؤال : چه متدی باید بنویسیم؟

ــ متد EH دکمه "ذخیره" : اکنون می‌خواهیم چنانچه کاربر بر روی دکمه "ذخیره" کلیک کرد، اطلاعات وارد شده در یک فایل متنی ذخیره شود. برای این منظور متد EH مناسبی را برای دکمه "ذخیره" می‌نویسیم.

در دستورات ابتدای متد EH، باید کنترل‌هایی بر روی محتوای کادرهای متنی صورت بگیرد. به عنوان مثال کادرهای متنی خالی نباشند و یا کاراکترهای فاصله از ابتدای آن حذف شده باشد.

همان طور که برای نوشتن مطلبی در دفتر خود، ابتدا دفتر را باز کرده و یک صفحه سفید آن را می‌آورید سپس مطلب را می‌نویسید و در پایان کار، دفتر را می‌بندید. برای ساختن فایل اطلاعاتی در کامپیوتر نیز باید همین سه عمل را انجام دهید :

۱ــ باز کردن فایل به منظور نوشتن اطلاعات

۲ــ نوشتن داده‌ها و اطلاعات درون فایل باز شده

۳ــ بستن فایل پس از اتمام کار

هر سه عملیات می‌تواند با متدهای مختلفی انجام شود. همچنین متدهایی وجود دارند که هر سه کار را پشت سرهم با یک دستور انجام می‌دهند. مثلاً برای ذخیره اطلاعات وارد شده در فرم شکل ۱ــ۵ ، از متد استاتیک WriteAllText()می‌توان استفاده کرد. این متد در کلاس File در حوزه نامی System.IO تعریف شده است. متد استاتیک WriteAllText() می‌تواند فایل جدیدی را ایجاد کند و اطلاعات را درون آن ذخیره کرده و سپس فایل را ببندد. اگر از قبل فایلی به همان نام وجود داشته باشد محتوای قبلی فایل پاک شده و متن جدید جایگزین می‌شود.

روش فراخوانی متد WriteAllText به صورت زیر است :

```
System.Io.File.WriteAllText(دادهٔ رشته‌ای، نام فایل)
```

با این توضیحات متد EH رویداد کلیک، دکمه "ذخیره" چنین خواهد بود :

```
Private void saveButton _ Click(object sender, EventArgs e)
{
    string errorTitle="خطا در ورود اطلاعات"

    userNameText.Text.Trim() ;
    if(userNameText.Length ==0)
    {
        MessageBox.show("لطفاً نام کاربری را وارد کنید",errorTitle);
        return;
    }
    PasswordText.Text.Trim();
    if (VerifiedText.Text.Length==0)
    {
        MessageBox.show("لطفاً کلمه عبور را وارد کنید",errorTitle);
        return;
    }
    verifiedText.Tent.Text.Trim();
    if (verifiedText.Text.Length==0)
    {
        MessageBox.show("لطفاً تکرار کلمه عبور را وارد کنید",eltiTrorre);
        return;
    }
    if (verifiedText. Text! = PasswordText. Text)
    {
        MessageBox.show("کلمه عبور با تکرار آن باید یکسان باشد ",errorTitle);
        return;
    }
    string textData = userName Text. Text +"," + PasswordText. Text;
    string fileName = "userlist.txt";

    System.Io.File.WriteAllText (fileName , textData);

    MessageBox.show("ثبت کاربر جدید" , ". اطلاعات به طور موفقیت‌آمیز ثبت شد");
    }
```

دقت کنید که متد EH رویداد کلیک دکمه ذخیره مقداری را برنمی‌گرداند ولی در صورت خالی‌بودن کادرهای متنی پس از نمایش یک پیام مناسب از دستور return برای خروج از متد و جلوگیری از اجرای خطوط بعدی استفاده شده است.

سؤال: متد WriteAllText کلاس File به صورت System.IO.File.WriteAllText استفاده شده است. در چه صورت می‌توان از آن به صورت File.WriteAllText استفاده کرد؟

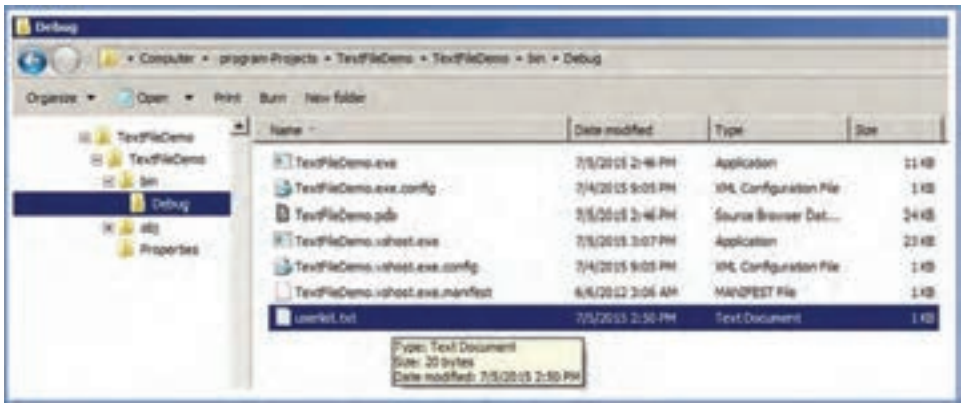
برنامه را اجرا کنید و اطلاعاتی را در فرم وارد کنید. اگر اطلاعات ناقص باشد خطایی متناسب با نقص اطلاعات مشاهده خواهید کرد (شکل ۲ــ۵).

شکل ۲ــ۵ــ نمایش خطای یکسان نبودن کلمهٔ عبور و تکرار آن

سعی کنید اطلاعاتی کامل و صحیح وارد نمایید. سپس بر روی دکمه "ذخیره" کلیک کنید (شکل ۳ ــ ۵).

شکل ۳ــ۵ــ نمایش پیام موفقیت آمیز بودن ثبت اطلاعات

اگر به فولدر حاوی پروژه مراجعه نمایید، باید فایل ایجاد شده را مشاهده کنید. برای این منظور ابتدا به مسیر ذخیره‌سازی پروژه که در ابتدای ساخت پروژه تعیین کرده‌اید، مراجعه کنید و سپس به شاخه فرعی Bin و بعد به شاخه Debug بروید(project directory\Bin\Debug). در لیست فایل‌ها مانند شکل ۴ــ۵ فایلی به نام userlist.txt را خواهید دید.



شکل ۴ــ۵ ــ لیست فایل‌های فولدر Debvg

برای مشاهده محتوای فایل userlist.txt روی نام آن، دو بار کلیک کنید. در این صورت محتوای فایل را در برنامه Notepad ویندوز مشاهده خواهید کرد :

شکل ۵ــ۵ ــ فایل userlist.txt

برنامه Notepad را ببندید و با کلیک بر روی دکمه "پاک کردن" محتوای کادرهای متنی را پاک کرده تا بتوانید اطلاعات جدیدی را وارد کنید. اطلاعات کاربر جدیدی را وارد و سپس ذخیره کنید (شکل ۶ــ۵).

شکل ۶ ــ ۵ــ ثبت اطلاعات کاربر جدید

اکنون به سراغ فایل متنی userlist.txt بروید و آن را باز کنید. در این صورت مشاهده خواهید کرد، که اطلاعات کاربر قبلی از بین رفته و اطلاعات کاربر جدید در آن قرار گرفته است.

شکل ۷ــ ۵ــ فایل

سؤال: چرا محتوای قبلی فایل از بین رفته است؟

در صورتی که بخواهید اطلاعات کاربر جدید به انتهای فایل موجود اضافه شود، باید از متد دیگری استفاده کنید که اطلاعات را در انتهای فایل موجود بنویسد. متد AppendAllText() چنین است :

اکنون متد EH دکمه ذخیره را به این صورت می‌نویسیم :

```
Private void saveButton _ Click(object sender, EventArgs e)
{
    ⋮
    string textData = userNameText.Text + ", " +passwordText.Text;
    string fileNeame = "userlist.txt";
    System.IO.File.AppendAllText(fileName, textData + "\r.\n");
    MessageBox.Show("ثبت کاربر جدید" , " اطلاعات بطور موفقیت‌آمیز ثبت شد");
}
```

رشته "\r\n" به عنوان کاراکتر خط جدید[1] در فایل متنی عمل کرده و سبب می‌شود پس از نوشتن اطلاعات کاربر در فایل، خط جدیدی ایجاد شود تا اطلاعات کاربر بعدی در ابتدای خط جدید نوشته شود.

برای مطالعه

کلمه عبور ۱۲۳۴۵۶ در صدر جدول بدترین ده کلمه عبور[2] در سال ۲۰۱۴ شناخته شده است. این کلمه عبور به راحتی قابل حدس است، بنابراین برای هیچ یک از حساب‌های کاربری خود از جمله بازی‌های رایانه‌ای آنلاین، چنین کلمه عبوری انتخاب نکنید، چون به وسیلهٔ هکرها، کلمه عبور شما حدس زده می‌شود و به راحتی امتیازات حساب خود را از دست می‌دهید.

متد AppendAllText قادر است اطلاعات را به انتهای فایل متنی اضافه[3] نماید. همچنین در صورتی که فایل متنی از قبل وجود نداشته باشد، آن را ایجاد کند.

سؤال: آیا می‌توان گفت که با وجود متد AppendAllText دیگر نیازی به متد WriteAllText نمی‌باشد؟

۱ـ New Line

۲ـ Top ten worst passwords

۳ـ Append

کار در کارگاه ۲: مشاهده اطلاعات درون فایل‌های متنی

مثال ۲ــ۵ : می‌خواهیم برنامه‌ای بنویسیم که اطلاعات درون فایل متنی را در یک کادر متنی نشان دهد.

شکل ۸ ــ ۵ــ فرم نمایش محتوای فایل

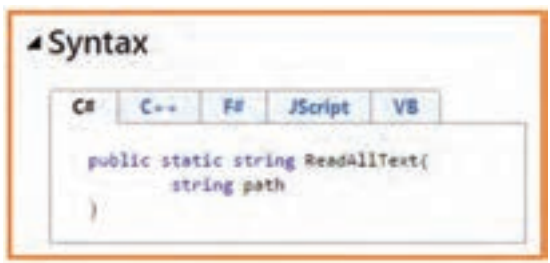
الگوریتم یا روش انجام کار : همانطور که برای خواندن یک کتاب، ابتدا کتابی را از قفسه برداشته، سپس آن را باز کرده، شروع به خواندن صفحه‌ای می‌کنیم و در انتها کتاب را بسته، در جای خود قرار می‌دهیم، برای خواندن فایل نیز باید عملیات زیر را انجام دهیم :

۱ــ باز کردن فایل به منظور خواندن اطلاعات

۲ــ خواندن داده‌ها و اطلاعات فایل باز شده

۳ــ بستن فایل پس از اتمام کار

خوشبختانه متدReadAllText هر سه عملیات بالا را انجام می‌دهد. بنابراین برای خواندن یک فایل متنی، از متد مزبور استفاده می‌کنیم. در شکل ۹ــ۵ خط عنوان متد ReadAllText، که از طریق MSDN قابل دسترسی است، نشان داده شده است.



شکل ۹ــ ۵ ــ ساختار متد ReadAllText در MSDN

متد ReadAllText از نوع استاتیک و با قابلیت دسترسی وسیع و قابل فراخوانی در هر برنامه‌ای تعریف شده است. متد مزبور دارای یک پارامتر ورودی از نوع رشته‌ای به نام path است که مسیر و نام فایلی که قرار است خوانده شود را مشخص می‌کند.

سؤال: با استفاده از شکل۹ــ۵ بگویید خروجی متد ReadAllText از چه نوعی است؟

با استفاده از این متد و با طی مراحل زیر می‌توانیم پروژه را انجام دهیم :

۱ــ **ایجاد یک پروژه ویندوزی** : وارد برنامه VS شوید و یک پروژه جدید از نوع WFA با نام ReadingFileDemo و در مسیر مشخصی بسازید.

۲ــ **تعیین ویژگی‌های فرم** :ویژگی‌های فرم را به صورت جدول۶ــ۵ تنظیم کنید.

جدول ۶ــ ۵ــ مقادیر ویژگی‌های شیء فرم

Form		
ویژگی		مقدار
Name		Form1
Text		مشاهده محتوای فایل
Size	Width	۳۰۰
	Height	۳۰۰

۳ــ **اضافه کردن یک برچسب بر روی فرم** : یک برچسب روی فرم اضافه کنید و ویژگی‌های آن را تنظیم کنید.

۴ــ **اضافه کردن یک کادر متنی برای دریافت نام فایل** : یک کادر متنی برای دریافت نام فایل به فرم اضافه کنید.

جدول۷ــ ۵ــ ویژگی کادر متن دریافت نام فایل

TextBox	
ویژگی	مقدار
Name	fileNameText

۵ــ اضافه کردن یک کادر متنی برای نمایش محتوای فایل : یک کادر متنی برای نمایش محتوای فایل مانند جدول ۸ــ۵ به فرم اضافه کنید. ویژگی‌های آن را مطابق سلیقه خود تنظیم کنید.

جدول ۸ــ۵ــ ویژگی‌های کادر متنی نمایش محتوای فایل

TextBox	
ویژگی	**مقدار**
Name	fileContentText
Multiline	True

۶ــ اضافه کردن یک دکمه : یک دکمه نیز به فرم اضافه کنید. خصوصیات آن را مانند آنچه در جدول ۹ــ۵ دیده می‌شود تنظیم کنید.

جدول ۹ــ ۵ــ ویژگی‌های دکمه نمایش محتوای فایل

Button	
ویژگی	**مقدار**
Name	showFileButton
Text	نمایش

۷ــ نوشتن متد EH رویداد دکمه : در متد EH دکمه "نمایش محتوای فایل" باید عملیات زیر را انجام دهیم :

۱ــ دریافت نام فایل از کادر متنی

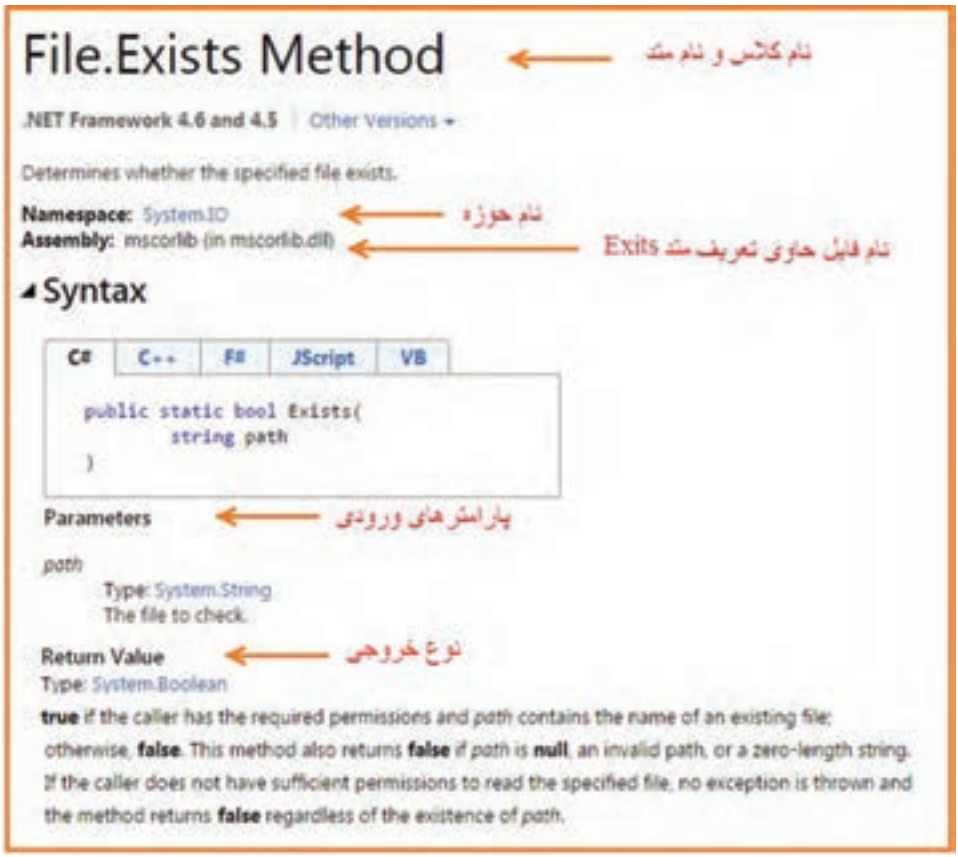
۲ــ اطمینان از وجود فایلی با نام وارد شده

۳ــ خواندن فایل

۴ــ بستن فایل

۲ــ۵ ــ استفاده از راهنمای برنامه

برای اطلاع از اینکه آیا فایلی که کاربر نام آن را وارد کرده است وجود دارد یا خیر، از متد Exist کلاس File استفاده می‌کنیم. فرض کنید که نحوه به کارگیری آن را نمی‌دانیم. بنابراین اگر در پنجره کدنویسی محیط VS قرار دارید و به اینترنت متصل هستید، نام متد را نوشته سپس با زدن کلید F۱ ، می‌توانید از طریق سایت MSDN اطلاعات کاملی در مورد آن به دست آورده و با این متد آشنا شوید.



شکل ۱۰ ــ ۵ــ راهنمای متد Exists

در شکل ۱۰ــ ۵ اطلاعات مربوط به متد Exists را مشاهده می‌کنید که در بالای صفحه نام کلاس و نام متد نوشته شده است. تمام متدهای NET دارای راهنمایی به صورت شکل بالا هستند. اگر عملکرد یک متد و یا پارامترهای آن را نمی‌دانید می‌توانید از راهنمای آن کمک بگیرید و سطح

معلومات خود را افزایش دهید. اگر به خط عنوان متد Exists یا به قسمت پارامترهای ورودی آن در شکل ۱۰ــ۵ توجه کنید، متد مزبور دارای یک پارامتر ورودی به نام path است که نام فایل را مشخص می‌کند و خروجی متد مقدار درست یا نادرست (نوع منطقی) است. اگر فایل وجود داشته باشد مقدار true و اگر فایل وجود نداشته باشد و یا اگر پارامتر path یک رشته تهی باشد، خروجی متد مقدار false را بر می‌گرداند.

برای خواندن فایل متنی نیز از متد ReadAllText با توجه به شرح زیر استفاده می‌کنیم :

شکل ۱۱ ــ ۵ــ راهنمای متد ReadAllText

سؤال: ورودی و خروجی متد از چه نوعی است؟ نحوه فراخوانی متد را برای هم کلاسی خود توضیح دهید.

اکنون متد EH برای دکمه را به صورت زیر می‌نویسیم :

```
Private void showFileButton _ Click(object sender, EventArgs e)
{
    string fileName = fileNameText.Text;
    if (System.IO.File. Exists(fileName))
      fileContentText.Text = System.IO.File.ReadAllText (fileName);
}
```

اکنون برنامه را اجرا کنید. نام فایلی را در کادر متنی وارد کنید و سپس بر روی دکمه "نمایش" کلیک کنید تا محتوای فایل در صورت وجود نمایش داده شود(شکل ۱۲ــ۵).

شکل ۱۲ ــ ۵ــ نمایش محتوای فایل Userlist.txt

خودآزمایی فصل پنجم

الف) درستی یا نادرستی هر عبارت را تعیین کنید.

۱ــ محتوای فایل‌های دودویی را می‌توان با برنامه‌های ویرایشگر به درستی مشاهده کرد.

۲ــ فایل‌ها درون حافظه‌های جانبی ذخیره می‌شوند.

۳ــ هنگام استفاده از متد WriteAllText، اگر از قبل فایلی به همان نام وجود داشته باشد، محتوای جدید جایگزین محتوای قبلی می‌شود.

ب) جاهای خالی را با عبارت مناسب پر کنید.

۴ــ اگر داده‌های موجود در برنامه به همان شکل که در حافظه قرار دارند، در فایل ذخیره شوند، فایل ایجاد شده از نوع ... است.

۵ــ برای نوشتن در فایل متنی از رشته "\r\n" به عنوان کاراکتر ..... استفاده می‌شود.

۶ــ متد AppendAllText قادر است اطلاعات را به ..... فایل متنی اضافه نماید.

۷ ــ خروجی متد ReadAllText از نوع ..... است.

۸ــ خروجی متد Exists از نوع ..... است.

۹ــ برای اطلاع از اینکه آیا فایلی که کاربر نام آن را وارد کرده است وجود دارد یا خیر، از متد ..... کلاس File استفاده می‌کنیم.

ج) به سؤالات زیر پاسخ دهید.

۱۰ــ چرا داده‌ها را در فایل ذخیره می‌کنیم؟

۱۱ــ دو تفاوت فایل متنی و دودویی را بنویسید.

۱۲ــ کاراکتر پایان فایل چیست؟

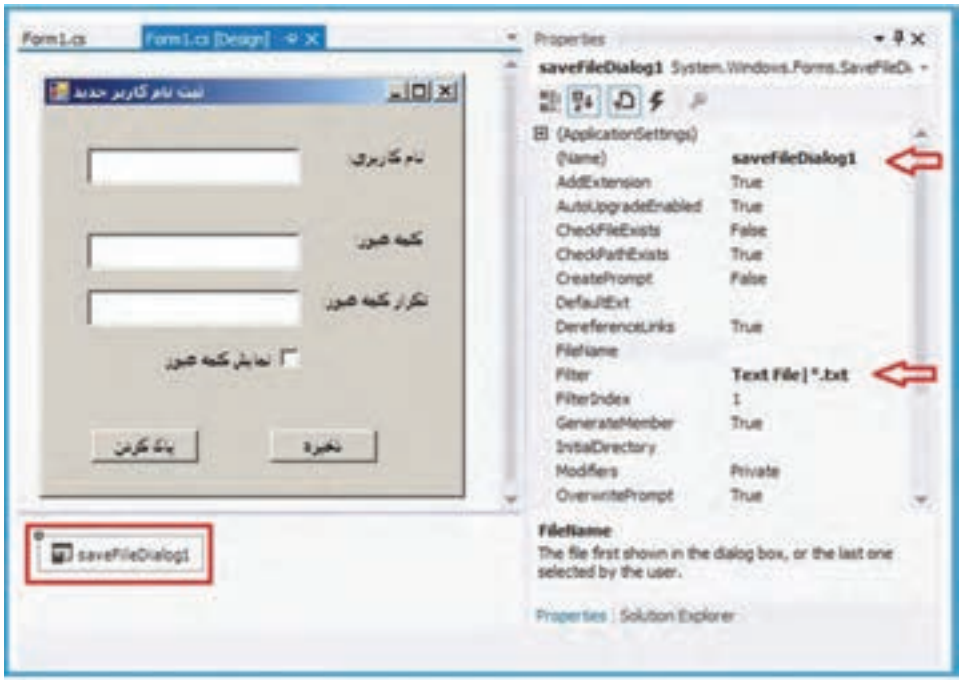
۱۳ــ دو روش دسترسی به اطلاعات درون فایل را با هم مقایسه کنید.

۱۴ــ کاربرد هر یک از متدهای زیر در کلاس File را برای هم‌کلاسی خود توضیح دهید.

الف ــ Exists　　ب ــ WriteAllText

ج ــ AppendAllText　　د ــ ReadAllText

۱ــ در مثال ۱، اگر بخواهیم نام فایل به وسیلهٔ کاربر انتخاب شود تا فایل در مسیر دلخواه کاربر ذخیره شود، می‌توانید از کادر محاوره‌ای DialogBox استفاده نمایید. شکل زیر اضافه شدن یک کادر محاوره‌ای ذخیرهٔ فایل را به فرم نشان می‌دهد. ویژگی Filter کادر محاوره‌ای برای نمایش فایل‌های متنی تنظیم شده است.



الف ــ تغییرات لازم برای این منظور را در کد برنامه ایجاد کنید (این تغییرات در راهنمایی قسمت ب آمده است)

ب ــ اگرکاربر فایل موجود را انتخاب کند اطلاعات به انتهای فایل اضافه شود.

راهنمایی : برای فراخوانی کادر محاوره‌ای که برای انتخاب نام و مسیر ذخیره فایل متنی، به کار می‌رود باید تغییراتی در برنامه اعمال کنیم. قطعه کد صفحهٔ بعد بخش پایانی متد EH رویداد کلیک دکمه "ذخیره" را نشان می‌دهد که تغییر پیدا کرده است :

```
Private void saveButton _ Click(object sender, EventArgs e)
{
   string errorTitle ="خطا در ورود اطلاعات" ;
      ⋮
   if (verifiedText.Text! = passwordText.Text)
   {
      MessageBox.show("کلمهٔ عبور؛ تکرار آن باید یکسان باشد",errorTitle);
      return;
   }
   string textData = userNameText.Text + ", "+ passwordText.Text;
   if (saveFileDialog1.showDialog() == DialogResult.ok)
   {
      System.IO.File.WriteAllText(saveFileDialog1.FileName, textData);
      MessageBox.Show("ثبت کاربر جدید" , "اطلاعات بطور موفقیت‌آمیز ثبت شد");
   }
}
```

اگر برنامه را اجرا کرده و پس از پرکردن اطلاعات کاربر، بر روی دکمه "ذخیره" کلیک کنید. می‌توانید فایلی را در مسیر دلخواه خود معین کنید تا اطلاعات وارد شده در فرم در آن ذخیره شود. اگر فایل موجود را انتخاب کنید، اطلاعات جدید جایگزین می‌شود. تغییری در برنامه ایجاد کنید تا اگر فایل موجود بود در انتهای آن اطلاعات کاربر را ذخیره کند.

۲ــ برنامه شماره ۲ــ۵ که مربوط به نمایش محتوای فایل‌های متنی است را مطابق لیست زیر تکمیل نمایید.

الف ــ به جای اینکه کاربر نام فایل را در کادر متنی تایپ کند، یک کادر محاوره‌ای باز کردن فایل (OpenFileDialog) ظاهر می‌شود و کاربر فایل دلخواه خود را برای خواندن باز می‌نماید.

ب ــ یک دکمه ذخیره به فرم اضافه شود تا تغییرات اضافه شده در متن، روی فایل اصلی ذخیره شود.

۳ــ برنامه ای بسازید که تعداد حروف فایل‌های متنی انتخاب شده را ذخیره کند و نمایش دهد.

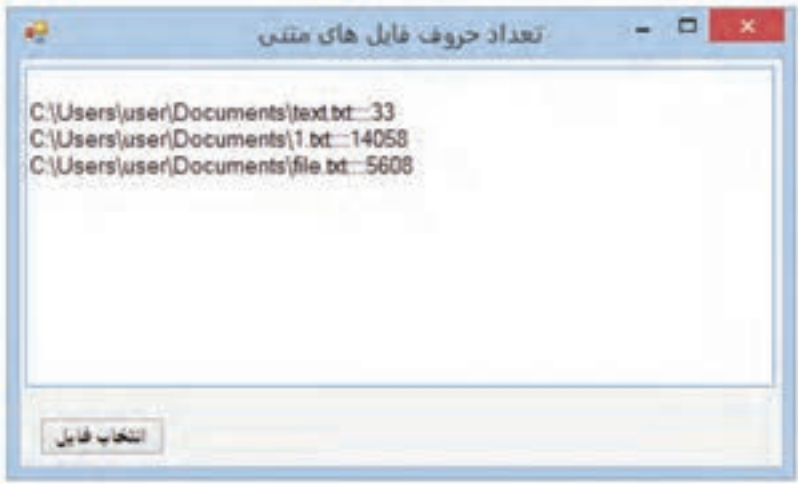
روش کار :

○ با زدن دکمه انتخاب فایل، یک کادر openFileDialog باز شود تا فایل متنی انتخاب شود.

o آدرس فایل و تعداد حروف در خط جدید از کادر متنی نگاشته شود. پس از آن متن کادر متنی در فایل length.txt ذخیره شود. نحوهٔ درج :

تعداد حروف : : : : نام فایل

o در متد load فرم، دستورهای لازم نوشته شود تا در هنگام شروع به اجرای برنامه، اطلاعات فایلlength (که در واقع اطلاعات تعداد حروف فایل‌های متنی انتخاب شده است) درکادر متنی نمایش داده شود.



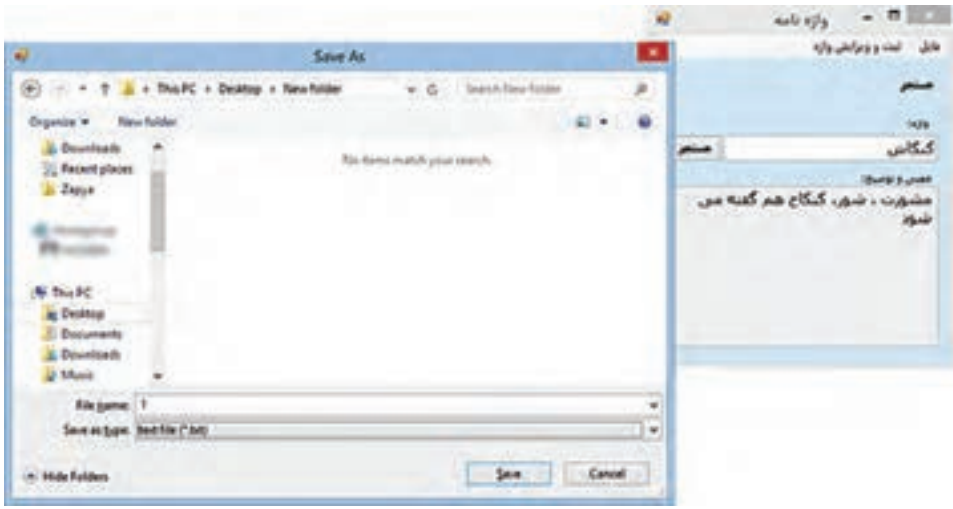
فعالیت

تکمیل پروژه

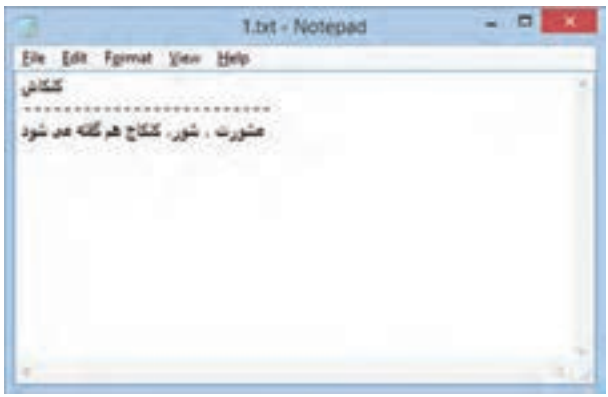
در این بخش از تکمیل پروژه، واژگان را در فایل ذخیره می‌کنیم.

مرحلهٔ پنجم :

در مراحل قبل گزینه "ذخیره جستجو" را برای ذخیره در فایل ایجاد کردیم و قسمتی از کد متد EH رویداد کلیک آن را نوشتیم که باز شدن کادر ذخیره فایل را انجام می‌داد. در این بخش ادامه کار روی فایل را خواهیم داشت.



با اضافه کردن دستوراتی به متد EHرویداد کلیلک منوی "جستجو و ذخیره" اطلاعات را در فایل به صورت شکل روبه‌رو ذخیره می‌کنیم.



متد EH رویداد کلیک گزینه ذخیره جستجو را به صورت زیر تغییر دهید.

```
private void saveMenuItem_Click(object sender, EventArgs e)
{
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.Filter = "text file|*.txt";
    if (sfd.ShowDialog() == DialogResult.OK)
        File.WriteAllText(sfd.FileName,searchWord.Text
        + " \r\n----------\r\n " +description.Text);
}
```

با شرط if (sfd.ShowDialog() == DialogResult.OK) مشخص می‌کنیم که اگر نتیجه کادر ذخیره فایل OK باشد، ذخیره فایل صورت گیرد. در این کادر دکمه Save نتیجه OK را برمی‌گرداند.

توسط متد WriteAllText، متن حاوی واژه و توضیح، در فایل متنی آدرس sfd.FileName یعنی آدرسی که کادر ذخیره فایل مشخص کرده است ذخیره می‌شود.

سؤال: چه تغییری در کد بالا ایجاد کنیم تا تمام واژه‌های جستجو شده و معنی آنها در یک فایل ذخیره شود. (واژه جدید به انتهای فایل اضافه شود)

واژگان و اصطلاحات انگلیسی فصل پنجم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Access	
۲	Append	
۳	Binary	
۴	Demo	
۵	Direct	
۶	Directory	
۷	Error	
۸	Exist	
۹	File	
۱۰	Open	
۱۱	Path	
۱۲	Read	
۱۳	Sequential	
۱۴	Stream	
۱۵	Temprory	
۱۶	Text	
۱۷	Verify	
۱۸	Write	

فصل ششم

پایگاه داده

برای اینکه بتوانیم داده‌های برنامه خود را ذخیره کنیم باید آن را در جایی از حافظه جانبی، نگهداری کنیم. یکی از راه‌های ذخیره، استفاده از فایل‌هایی است که برنامه می‌سازد و داده‌ها را در آن ذخیره می‌کند. اما این روش زمانی که اطلاعات افزایش یافت و نیاز به مدیریت بیشتر اطلاعات و سرعت بیشتر باشد، ناکارآمد است. بنابراین ابزار دیگری نیاز است تا داده‌ها را در حجم زیاد بتواند ذخیره کند و مدیریت قدرتمند و متمرکزی را به همراه سرعت در کار ذخیره و بازیابی، فراهم نماید. راه حل این مهم، استفاده از پایگاه داده است.

در این فصل آنچه اهمیت دارد این است که چگونه بین برنامه خود، با پایگاه داده ارتباط برقرار نماییم؟

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ از طریق برنامه به زبان C # اتصال بین پایگاه داده و برنامه ایجاد نماید.

۲ــ فایل پایگاه داده را با استفاده از SQL Server در محیط VS ایجاد کند.

۳ــ از ابزارهای نمایش داده‌ها بردر روی فرم استفاده کند.

۴ــ برنامه‌های کاربردی مبتنی بر پایگاه داده بنویسد.

۱ــ۶ــ پایگاه داده چیست

پایگاه داده[۱] حجم انبوهی از اطلاعات است که به طور منظم و سازماندهی شده بر روی حافظه‌های جانبی به صورت فایل نگهداری می‌شود. اطلاعات درون یک پایگاه داده به سادگی و با سرعت، قابل دستیابی است. معمولاً در یک پایگاه داده اطلاعات در یک یا چند جدول قرار می‌گیرند[۲]. هر جدول شامل اطلاعاتی در مورد یک موضوع است. مثلاً پایگاه داده یک مدرسه که اطلاعات درسی دانش‌آموزان را نگهداری می‌کند، می‌تواند شامل چندین جدول باشد. اطلاعات شناسنامه‌ای دانش‌آموزان در یک جدول و نمرات آنها در جدول دیگری نگهداری می‌شود. در جدول دانش آموزان، اطلاعاتی نظیر نام، نام خانوادگی، شمارهٔ ملی و آدرس مربوط هر یک از دانش آموزان مدرسه نگهداری می‌شود. هر سطر از این جدول یک رکورد دانش آموزی است و هر یک از اجزای تشکیل‌دهندهٔ این رکورد، فیلد[۳] نامیده می‌شود(شکل ۱ــ۶).

شکل ۱ــ۶ــ جدول اطلاعات دانش‌آموز

سؤال: با توجه به شکل ۱ــ۶، رکورد دانش آموز، از چند فیلد تشکیل شده است؟

بنابراین یک پایگاه داده کاربردی، از یک یا چند جدول تشکیل شده است و هر جدول نیز از

---

۱ – Data Base

۲ ــ Table

۳ــ Field

تعدادی رکورد و هر رکورد از تعدادی فیلد.

سؤال: جدول معرفی دروس یک رشته از چه فیلدهایی تشکیل می‌شود؟

تمرین : به جدول شکل ۱ــ۶ فیلد و دو رکورد دیگر اضافه کنید.

برنامه‌ای که اطلاعات درون یک پایگاه داده را سازماندهی و مدیریت می‌نماید، سیستم مدیریت پایگاه داده[۱] یا به اختصار DBMS نامیده می‌شود. مزیت استفاده از DBMS برای نگهداری و دسترسی به اطلاعات را در درس بانک اطلاعاتی خوانده‌اید. برنامه ACCESS یک نمونه از DBMS است که همراه مجموعه Microsoft Office عرضه می‌شود. برنامه Access برای نگهداری اطلاعات و داده‌های شرکت‌ها و مؤسسات کوچک مناسب است.

برای مطالعه

چنانچه تعداد رکوردها و حجم اطلاعات زیاد باشد، باید از DBMS پیشرفته‌تری مانند Microsoft SQL Server و یا اوراکل[۲] که بسیار پیشرفته و گران قیمت هستند، استفاده کرد. شرکت مایکروسافت نرم‌افزار SQL Server را در نسخه‌های مختلفی با سطوح قابلیت و توانایی‌های متفاوتی عرضه کرده است. در بین آنها، نسخه Express به صورت رایگان از سایت مایکروسافت قابل دانلود است. همچنین نسخه کم حجمی از Express با نام LocalDB منتشر شده است که نیاز به تنظیم‌های خاص Express را ندارد[۳].

نسخه Sql Express به همراه نصب ویژوال استودیو می‌تواند نصب شود. همچنین در صورت نیاز می‌توان آن را به راحتی از وبسایت مایکروسافت دانلود نمود و نصب کرد. در ادامه این فصل با استفاده از امکان فراهم شده توسط این نرم‌افزار به ایجاد یک پایگاه داده خواهیم پرداخت.

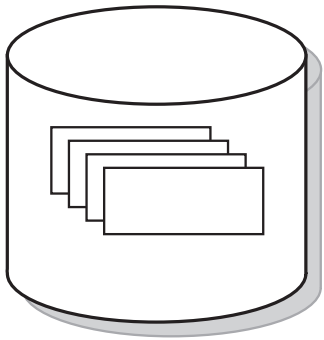
۱ــ Data Base Management System

۲ــ Oracle. www.oracle.com

۳ــ نسخه Sql Compact Edition. نیز از طرف مایکروسافت ارائه شده است اما بعد از نسخه ۲۰۱۲ سایر نسخه‌های ویژوال استودیو از آن پشتیبانی نمی‌کنند.

نکته

شکل نمادین ۲ــ۶ یک پایگاه داده را نشان می‌دهد. هر پایگاه داده دارای یک نام است که از تعدادی جدول به نام‌های مختلف تشکیل شده است.

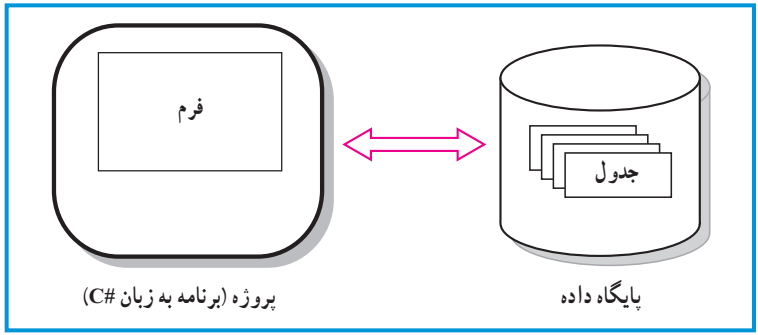


شکل ۲ــ۶ ــ پایگاه داده

۲ــ۶ــ ایجاد پایگاه داده توسط Visual studio و استفاده از آن

در این بخش می‌خواهیم از طریق یک پروژه WFA که در داخل VS ایجاد می‌کنیم به یک پایگاه داده متصل شویم و اطلاعاتی را درون آن قرار داده و یا اطلاعات موجود آن را مشاهده نماییم. در این فصل ابتدا یک نمونه پایگاه داده برای نگهداری اطلاعات دفتر تلفن ایجاد می‌کنیم.

شکل ۳ــ۶ به‌طور کلی نشان می‌دهد که می‌توان اطلاعات درون یک پایگاه داده را در پروژه نرم‌افزار کاربردی استفاده کرد و یا ویرایش کرد. به عبارت دیگر از طریق کنترل‌های موجود در یک فرم، می‌توان داده‌های درون پایگاه داده را مشاهده کرد. همچنین داده‌هایی که کاربر در روی فرم وارد می‌کند و یا اطلاعات دستکاری[۱] شده، می‌تواند در پایگاه داده، ذخیره شود (شکل ۳ــ۶).



شکل ۳ــ۶ــ ارتباط پروژه نرم‌افزار با پایگاه داده

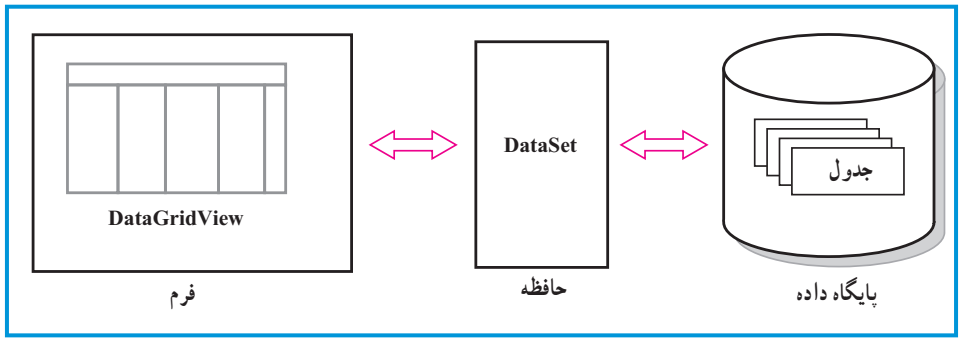
۱ــ Manipulated

مشکل بزرگی که معمولاً در کار با پایگاه داده وجود دارد این است که تاکنون همه عملیات بر روی اطلاعات درون متغیرها انجام می‌شد که آنها در حافظه Ram کامپیوتر قرار داشتند و سرعت دسترسی به آنها بسیار بالا بود. اما هنگامی‌که داده‌ها در پایگاه داده قرار دارند و در حافظه‌های جانبی ذخیره شده‌اند، با وجود سرعت بالای کار آن، در مقایسه با داده‌های موجود در Ram، کار با داده‌ها سرعت کمتری خواهد داشت. برای بالا بردن سرعت عملیات و افزایش کارایی سیستم، معمولاً بخشی از اطلاعات درون یک پایگاه داده، به داخل حافظه RAM آورده شده و سپس در درون فرم نمایش داده و یا دستکاری می‌شود.

بنابراین عملیات لازم برای دسترسی به اطلاعات درون یک پایگاه داده، به دو مرحله تقسیم می‌شود :

۱ــ ابتدا اطلاعات موردنظر از داخل پایگاه داده، به حافظه RAM کامپیوتر منتقل می‌شود به‌طوری که قابل مدیریت باشد. این عمل به وسیلهٔ ابزاری به نام DataSet انجام می‌شود.

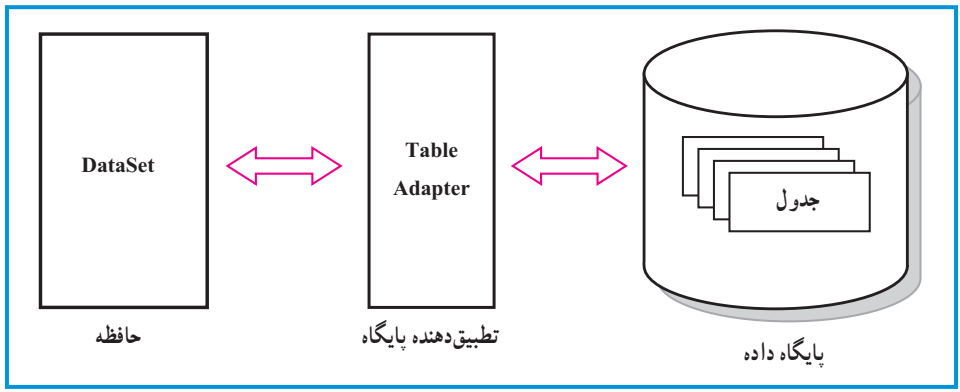
۲ــ سپس باید از ابزاری برای نمایش اطلاعات در روی فرم مورد استفاده قرار گیرد. هنگامی‌که بخواهیم اطلاعات یک جدول را در روی فرم نمایش دهیم از ابزار DataGridView استفاده می‌کنیم. اما برای نمایش اطلاعات یک یا چند فیلد از ابزارهای دیگری که تاکنون با آنها آشنا شده‌اید، نظیر کنترل‌های برچسب یا جعبه متنی می‌توان استفاده کرد (شکل ۴ــ۶).



شکل ۴ــ۶ــ پروژه (برنامه به زبان #C)

با توجه به اینکه انواع مختلفی از پایگاه داده وجود دارد و سیستم مدیریت هر یک متفاوت است، برای اینکه نیاز به انواع مختلف DataSet نباشد و برنامه کاربردی مستقل از نوع پایگاه داده عمل نماید از یک واسطه به نام تطبیق‌دهنده جدول TableAdapter استفاده می‌شود. بنابراین برای هر

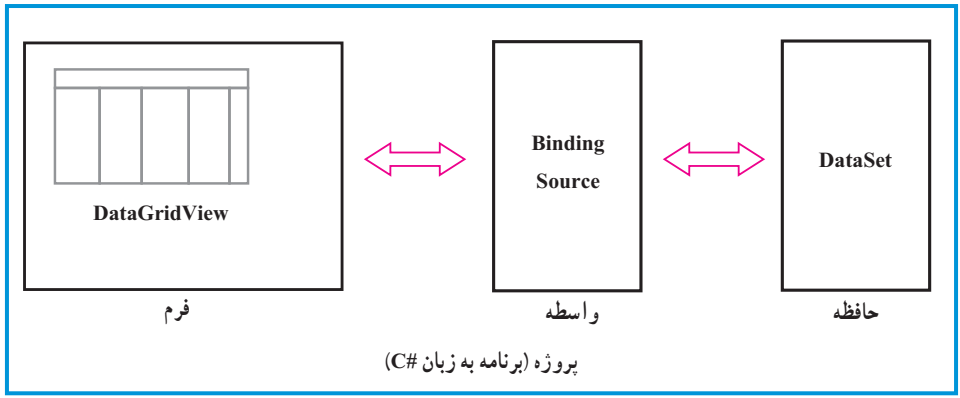
پایگاه داده، یک تطبیق‌دهنده مخصوص طراحی می‌شود که از آن برای انتقال اطلاعات به DataSet استفاده شود (شکل ۵ــ۶).



شکل ۵ــ۶ــ ارتباط DataSet و حافظه RAM

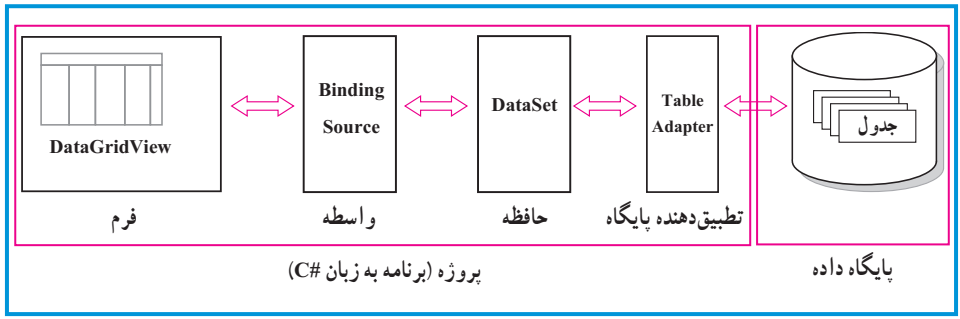
اکنون به سراغ مرحلهٔ دوم می‌رویم :

برای انجام قدم دوم یعنی نمایش داده‌ها در روی فرم، از یک واسطه دیگری به‌نام BindingSource استفاده می‌شود. به وسیلهٔ این واسطه، کنترل‌های نمایشی در روی فرم به‌راحتی می‌توانند به بخشی از اطلاعات موجود در حافظه (DataSet) دسترسی داشته باشند (شکل ۶ــ۶).



شکل ۶ــ۶ــ ارتباط فرم با DataSet

برای هر یک از عملیات بالا کلاس و یا ابزاری تهیه شده است که از آنها در برنامه استفاده می‌کنیم (شکل ۷ــ۶).



شکل ۷ــ۶ــ ارتباط پروژه با پایگاه داده

آن‌طور که به نظر می‌رسد از دیدن تصویر و توضیحات ارائه شده، کمی‌ناامید و مأیوس[1] شده‌اید. ناامید نباشید. در کتابخانه NET. مجموعه‌ای از کلاس‌های قدرتمندی برای اتصال با پایگاه داده و انجام عملیات مختلف بر روی داده، تحت عنوان ADO.NET تعریف شده است. در محیط VS با استفاده از ابزارهای تصویری و مرحله به مرحله می‌توانید تمام اشیای لازم برای اتصال به یک پایگاه داده را به‌دست آورید بدون اینکه لازم شود دستوراتی را بنویسید.

۳ــ۶ــ انواع کلاس‌ها و ابزارهای کار با پایگاه داده

۱ــ۳ــ۶ــ ابزار DataGridView : یک ابزار واسطه گرافیکی است که برای نمایش و به‌روز رسانی رکوردهای اطلاعاتی استفاده می‌شود.

۲ــ۳ــ۶ــ ابزار BindingSource : کلاسی است که داده‌های نمایش داده شده بر روی فرم را به منبع آن اتصال می‌دهد.

۳ــ۳ــ۶ــ ابزار DataSet : کلاسی است برای ذخیره داده‌های پایگاه داده در حافظه. شیء ساخته شده از این کلاس می‌تواند اطلاعات یک یا چند جدول را ذخیره نماید.

۴ــ۳ــ۶ــ ابزار TableAdapter : کلاسی است که ارتباط بین پایگاه داده با نرم‌افزار کاربردی را مهیا می‌کند.

اکنون برای آشنایی با ابزارهای توضیح داده شده، مثالی را انجام می‌دهیم :

۱ــ Disappointed

کار در کارگاه: ساخت یک دفتر تلفن

مثال ۱ : می‌خواهیم یک دفتر تلفن بسازیم که بتوان اطلاعات دوستان و آشنایان را در آن ذخیره کنیم.

برای آنکه یک نرم‌افزار متصل به پایگاه داده بسازیم مراحل زیر را باید دنبال کنیم :

۱ــ ایجاد الگوریتم و نقشه کار مناسب

۲ــ طراحی واسط کاربری مناسب

۳ــ طراحی پایگاه داده مورد نیاز

۴ــ ایجاد رابطه بین نرم‌افزار و پایگاه داده

۵ــ تکمیل کدها و برنامه

۶ــ آزمایش و رفع اشکال برنامه

توجه شود که برخی از این مراحل می‌تواند بنابر مقتضیات کار، جابه‌جا شود. به‌عنوان مثال گاهی ابتدا پایگاه داده ساخته می‌شود و بعد کنترل‌های مربوط به واسط کاری اضافه می‌شوند.

الگوریتم و روش انجام کار : ابتدا یک واسط کاربری مناسب برای نمایش و درج افراد و شماره تماس آنها طرح ریزی می‌کنیم .نمونه کار به صورت شکل ۸ــ۶ است :

شکل ۸ــ۶

برای نگهداری از این اطلاعات از یک پایگاه داده استفاده می‌کنیم. می‌توانید با برنامه Access پایگاه را ایجاد کنید و یا مانند مثال از sql server استفاده نمایید. پس از ایجاد پایگاه، به سراغ ابزارهای اتصال به پایگاه داده رفته و پس از آن، کنترل‌های نمایش اطلاعات را روی فرم قرار داده و یک واسط کاربری مناسبی را طراحی می‌نماییم. همان طور که دیدید ابتدا پایگاه داده ساخته شد و سپس کنترل مربوط اضافه شد. دلیل این تصمیم‌گیری را در ادامه خواهید دید.

اکنون برای ساخت اولین پروژه بانک اطلاعاتی به شرح زیر عمل می‌کنیم :

۱ــ *ایجاد واسط کاربری مورد نیاز :* یک پروژه ویندوزی WFA به نام AddressBook در مسیر دلخواه خود ایجاد کنید. با توجه به طرح ریزی انجام شده فعلاً کنترل دیگری نیاز نیست که اضافه شود.

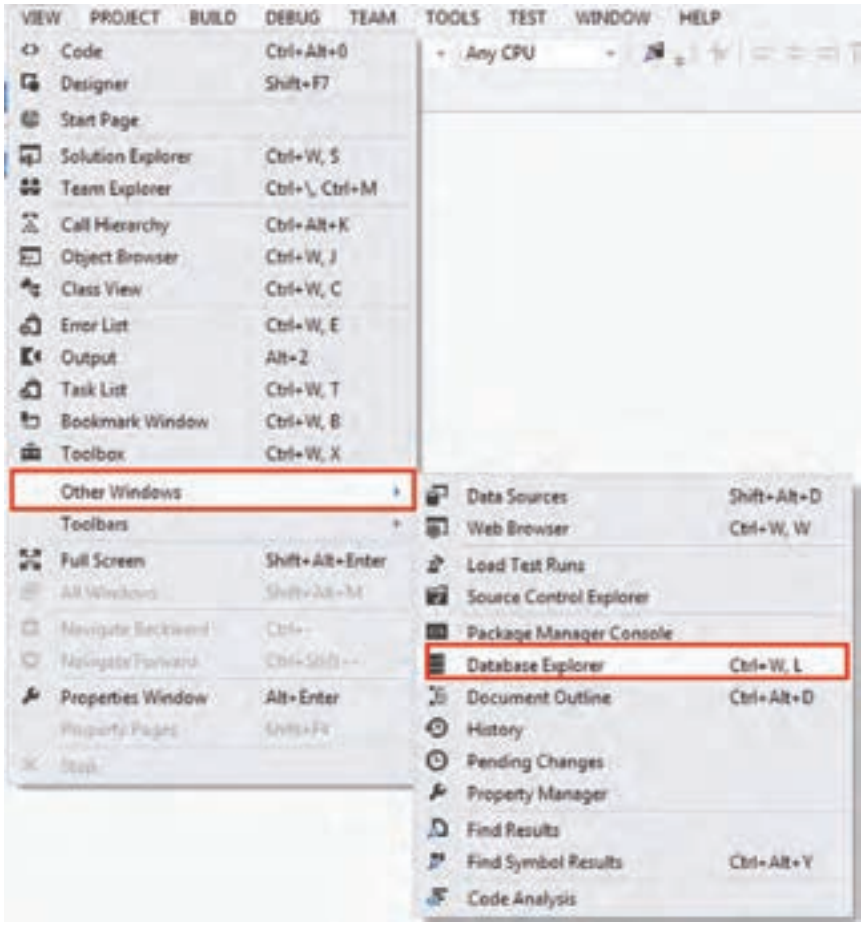
۲ــ *طراحی و ایجاد پایگاه داده مورد نیاز :* یک پایگاه داده‌ها برای موضوع مسئله، طراحی می‌کنیم. البته در این مثال به جهت سادگی، یک پایگاه داده به نام myDb ایجاد کرده که تنها شامل یک جدول اطلاعاتی به نام numbers است (جدول ۱ــ۶).

جدول ۱ــ۶ــ طرح جدول اطلاعاتی برای دفتر تلفن

نام و نام خانوادگی	شهر محل اقامت	شماره تلفن ثابت	شماره تلفن همراه

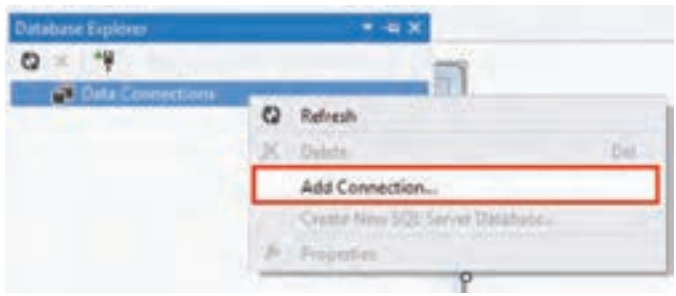
۳ــ پنجره مرورگر پایگاه داده (Database Explorer) را از منوی View بیاورید[۱] (شکل ۹ــ۶).

۱ــ در نسخه‌های غیر از express می‌توانید از پنجره server explorer نیز استفاده کنید.



شکل ۹ــ۶ــ اجرای فرمان نمایش پنجره مرورگر پایگاه داده

۴ــ در پنجره مرورگر پایگاه داده‌هاداده (Database Explorer)، کلیک راست کرده و گزینه Add Connection… را انتخاب کنید (شکل ۱۰ــ۶).



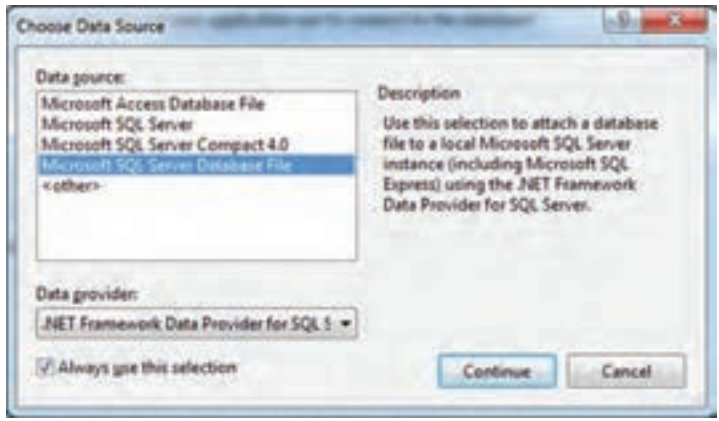
شکل ۱۰ــ۶ــ اتصال اضافه‌کردن

۵ــ در پنجره Add connection باید مشخصات پایگاه داده‌ها را مشخص کنیم (شکل ۱۱ــ۶).

۶ــ در قسمت Data Source نوع پایگاه داده را معین می‌کنیم. با کلیک بر روی دکمه Change در شکل ۱۱ــ۶ می‌توانید نوع پایگاه داده مورد نظر خود را انتخاب کنید.

شکل ۱۱ــ۶ــ کادر اضافه‌کردن اتصال

۷ــ در این مثال پایگاه داده‌ای از نوع Microsoft Sql Server DataBase File را معین می‌کنیم (شکل ۱۲ــ۶).



شکل ۱۲ــ۶ــ انتخاب منبع داده

۸ــ پس از تعیین نوع پایگاه داده و زدن دکمه continue، باید مسیر و نام پایگاه را مشخص کنیم. برای آنکه فایل در محل پیش‌فرض (پوشه documents) ایجاد شود در کادر Database file name نام دلخواه را برای پایگاه داده می‌نویسیم، اما اگر می‌خواهید در مکان دیگری فایل را ذخیره نمایید، برای این منظور می‌توانید بر روی دکمه Browse کلیک کنید. در اینجا فایل در محل پیش‌فرض ایجاد شده است (شکل ۱۳ــ۶).

شکل ۱۳ــ۶ــ نام‌گذاری فایل پایگاه داده

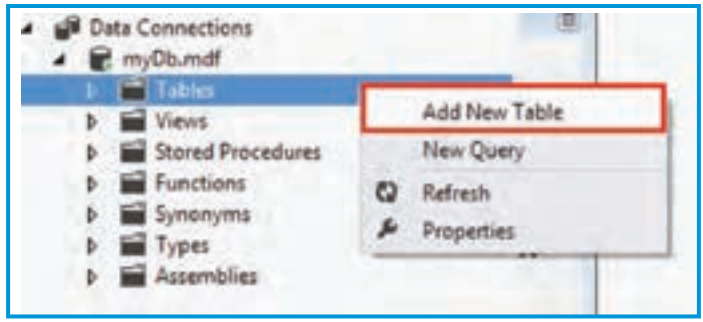
همچنین اگر فایل پایگاه داده را قبلاً ساخته‌اید می‌توانید در این مرحله، آن فایل را انتخاب کنید. در این صورت فایل جدید ایجاد نخواهد شد و از فایل انتخاب شده، استفاده خواهد شد.

۹ــ با زدن دکمه ok، پیامی‌نمایش داده خواهد شد که می‌گوید در این فایل پایگاه داده وجود ندارد. آیا می‌خواهید آن را ایجاد کنید؟ شما با زدن دکمه Yes فایل را خواهید ساخت.

۱۰ــ پس از ساخته شدن فایل در پنجره مرورگر پایگاه داده خواهیم دید یک اتصال به پایگاه داده جدید، ایجاد شده است (شکل ۱۴ــ۶).

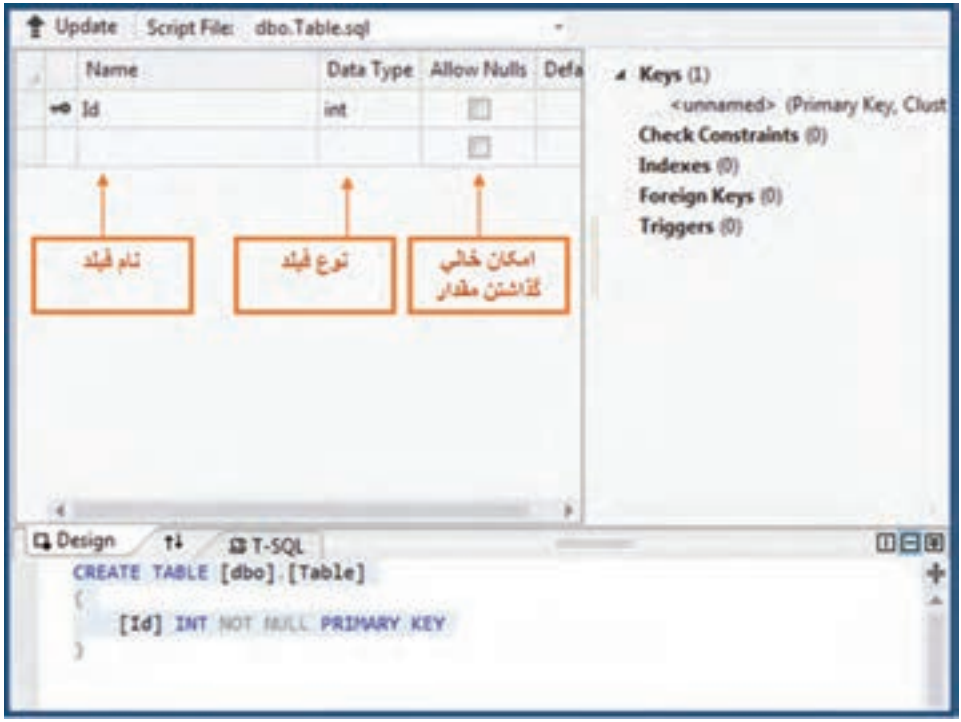
شکل ۱۴ــ۶ــ نمایش اتصال به پایگاه داده

۱۱ــ در مرحلهٔ بعد روی مثلث کنار نام فایل کلیک کرده تا زیر شاخه‌های آن دیده شود. سپس بر روی شاخه Tables کلیک راست کرده و مانند شکل ۱۵ــ۶ گزینه Add New Table را انتخاب کنید.



شکل ۱۵ــ۶ــ اضافه‌کردن یک جدول جدید

۱۲ــ در پنجره‌ای که ظاهر می‌گردد می‌توانید نام و مشخصات جدول جدید را وارد کنید (شکل ۱۶ــ۶).



شکل ۱۶ــ۶ــ طراحی جدول

برای آنکه نوع یک فیلد را مشخص کنید باید اطلاعاتی را دربارهٔ انواع داده‌ای در Sql server بدانید. از آنجا که در اینجا هدف آشنایی اولیه شما با Sql server می‌باشد. تعداد محدودی از این انواع را در جدول ۲ــ۶ مرور می‌کنیم.

جدول ۲ــ۶

نوع	توضیح
char	برای فیلدهای متنی غیر یونیکد درنظر گرفته می‌شود. بنابراین اگر متنی که می‌خواهید وارد کنید انگلیسی است از این نوع استفاده کنید. همچنین این نوع مناسب برای رشته‌هایی با طول معین می‌باشد.
nchar	برای فیلدهای متنی یونیکد استفاده می‌شود. بنابراین برای وارد کردن متنی که حروف فارسی دارد این نوع استفاده می‌شود. این نوع نیز برای رشته‌های با طول معین، مناسب است.
nvarchar	برای فیلدهای متنی یونیکد این نوع در نظر گرفته می‌شود. مقدار بایت در نظر گرفته شده برای هر فیلد بستگی به تعداد حرف متن ورودی دارد. این نوع برای رشته‌های متنی با طول نا معین مناسب است.
int	برای مقادیر عددی صحیح، از این نوع استفاده می‌شود.

در اکثر مواقع برای فیلدهای متنی از نوع nvarchar استفاده می‌شود. زیرا ما از نوشتار فارسی استفاده می‌کنیم و همچنین در بیشتر موارد رشته متنی ورودی، طول ثابتی ندارد.

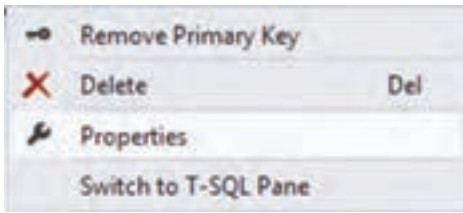
همچنین مقدار حداکثر کاراکتر درنظر گرفته شده در فیلدهای متنی، جلوی نوع و در درون پرانتز نوشته می‌شود. به‌عنوان مثال اگر برای یک فیلد متنی حداکثر ۲۰ حرف در نظر گرفته‌اید نوع آن به صورت nvarchar(20) در نظر گرفته می‌شود.

در لیست شماره تلفن‌ها، فیلدهای جدول ۳ــ۶ را برای جدول در نظر بگیرید.

جدول ۳ــ۶ــ تعیین نام و نوع فیلدها

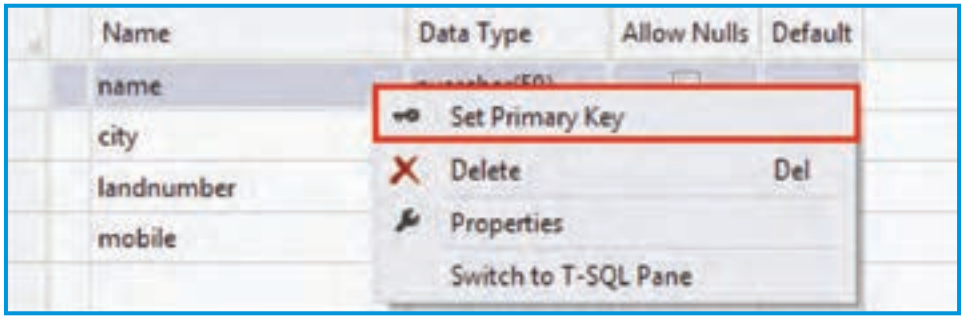
نام فیلد	نوع	کاربرد
name	nvarchar(50)	نام شخص
city	nvarchar(30)	شهر
landnumber	char(8)	شمارهٔ منزل
mobile	char(11)	شمارهٔ همراه

پس از وارد کردن فیلدها می‌توانید فیلد id را که در ابتدا ایجاد شده بود با کلیک راست روی آن و زدن delete حذف کنید (شکل ۱۷ــ۶).



شکل ۱۷ــ۶ــ منوی زمینه فیلدها

سپس می‌توانید با کلیک راست روی فیلد name، گزینه Set Primary Key را بزنید. با این کار این فیلد به‌عنوان کلید اصلی شناخته خواهد شد. البته شما می‌توانستید فیلد id را از ابتدا تبدیل به name کنید (شکل ۱۸ــ۶).

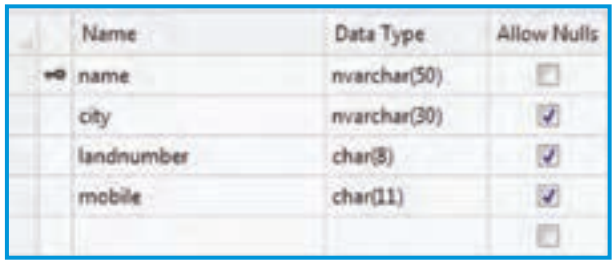


شکل ۱۸ــ۶

سؤال: آیا فیلد مجازی را می‌توان به عنوان کلید اصلی ایجاد و معرفی کرد؟

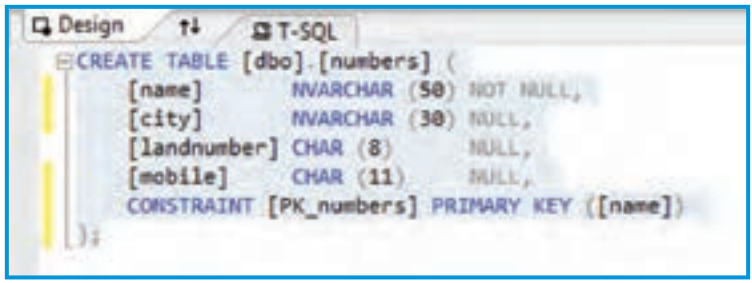
سؤال: آیا از ترکیب فیلدها می‌توان به عنوان کلید اصلی استفاده کرد؟

فیلدهای وارد شده را در شکل ۱۹ــ۶ می‌توانید مشاهده کنید.



شکل ۱۹ــ۶ــ فیلدهای ایجادشده در محیط طراحی

۱۳ــ پس از تعیین فیلدها نام جدول را مشخص می‌کنیم. برای این کار در کد پایین پنجره طراحی، نام جدول را numbers تعیین کنید (شکل ۲۰ــ۶).

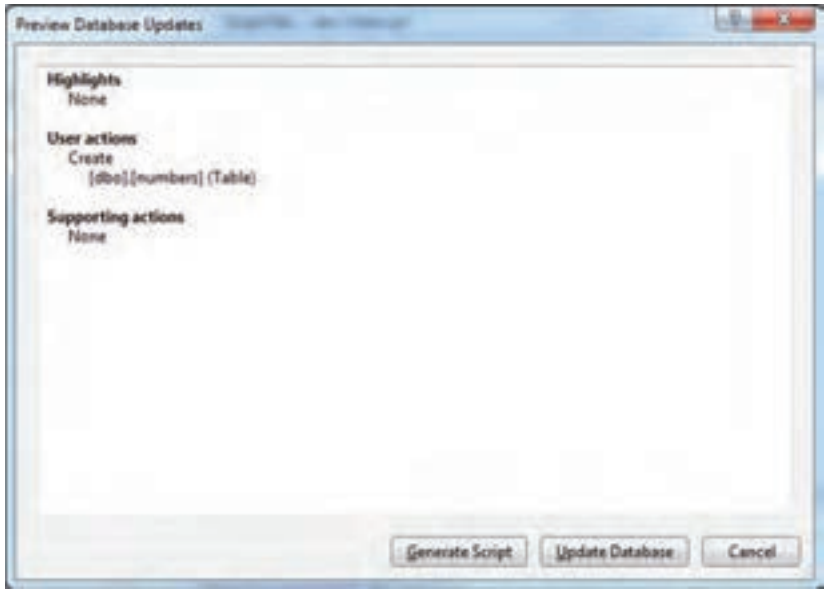


شکل ۲۰ــ۶ــ کد SQL برای ایجاد جدول

۱۴ــ اکنون نوبت بروزرسانی پایگاه داده و ثبت تغییرهای انجام شده است. ابتدا دکمه update در بالای کادر طراحی جدول را کلیک کنید (شکل ۲۱ــ۶).

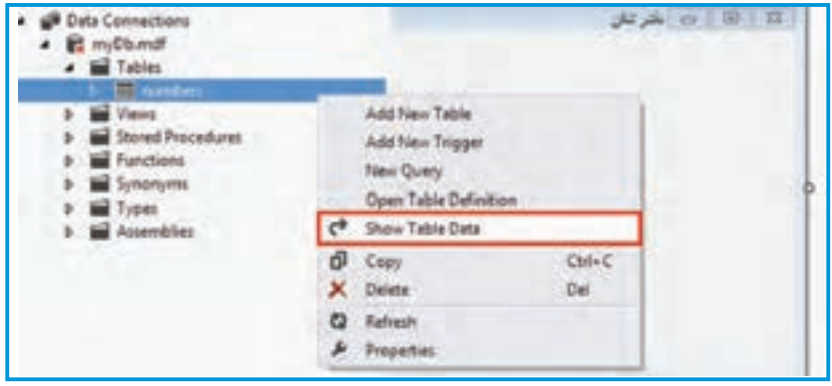
شکل ۲۱ــ۶ــ دکمه update

سپس با زدن دکمه Update Database در پنجره باز شده، جدول در پایگاه ثبت خواهد شد (شکل ۲۲ــ۶).



شکل ۲۲ــ۶ــ کادر پیش‌نمایش به‌روزرسانی پایگاه داده

۱۵ــ پس از ثبت تغییرات، برای اینکه بتوان تعدادی رکورد در جدول وارد کرد، به پنجره مرورگر پایگاه داده[۱] باز گردید و روی نام جدول numbers کلیک راست کرده و گزینه Show Table Data را انتخاب کنید (شکل ۲۳ــ۶).



شکل ۲۳ــ۶ــ اجرای فرمان نمایش داده‌های جدول

۱ــ DataBase Explorer

۱۶ــ با کلیک بر روی گزینه Show Table Data، می‌توانید عمل ورود اطلاعات را انجام دهید. علامت * در ابتدای سطر، نشانه رکورد جدید است. از سطر جدید، شروع کرده و اطلاعات خود و چند تن از دوستان خود را بنویسید. لزومی‌ندارد اطلاعات تمام افراد را وارد کنید، بعداً نیز می‌توانید اطلاعاتی را مانند شکل ۲۴ــ۶ به جدول اضافه کنید. آنچه در اینجا آورده شده فقط برای نمونه آورده شده است اما شما سعی کنید اطلاعات با معنی وارد کنید.

	name	city	landnumber	mobile
	حسن نادری	تهران	11111111	09123456789
	حسن نوروزیان	NULL	NULL	NULL
	محمد قاسمی	شیراز	2222222	09121111111
▶*	NULL	NULL	NULL	NULL

شکل ۲۴ــ۶ــ الف ــ رکوردهای واردشده در جدول

۱۷ــ *طراحی واسط کاربری و اتصال به پایگاه داده :* پس از ایجاد پایگاه و جدول اطلاعاتی می‌توانیم به سراغ طراحی واسطه کاربری (فرم) برویم. بنابراین پنجره طراحی فرم را باز کرده تا با اضافه کردن واسطه‌ها اتصال به پایگاه داده را برقرار کرده و اطلاعات مورد نظر را روی فرم نمایش دهیم.

۱۸ــ ویژگی‌های فرم را طبق جدول ۴ــ۶ تنظیم کنید :

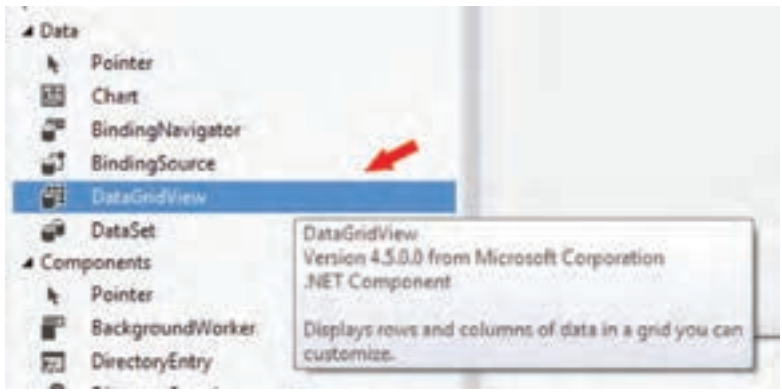
جدول ۴ــ۶ــ ویژگی‌های فرم

شیء Form		
ویژگی		مقدار
Name		Form1
Text		دفتر تلفن
RightToLeft		Yes
Size	Width	600
	Height	300

۱۹ــ برای اینکه بتوان اطلاعات موجود در یک جدول پایگاه داده را، به صورت یک جدول نمایشی در روی فرم مشاهده کرد از ابزار DataGridView استفاده می‌کنیم. برای این منظور به جعبه ابزار Data مراجعه کرده و کنترل DataGridView را با ویژگی‌های جدول ۵ــ۶ به فرم اضافه کنید.

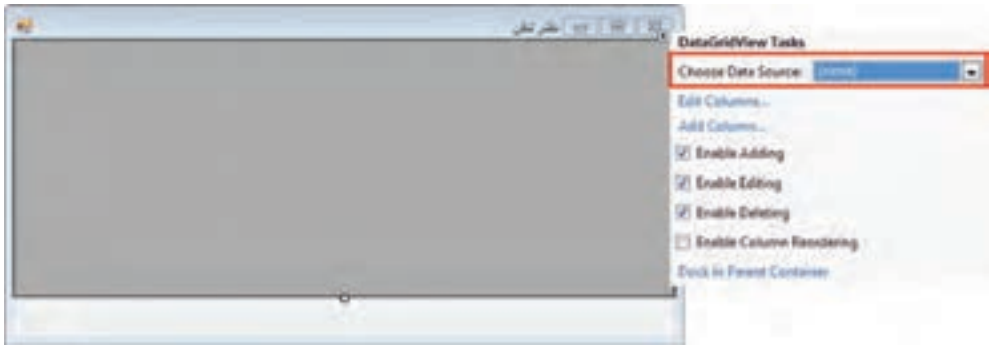
جدول ۵ــ۶ــ ویژگی‌های Data Gridview

شیء DataGridView		
ویژگی		مقدار
Name		dataGridView1
RightToLeft		Yes
Dock		top
Size	Height	227



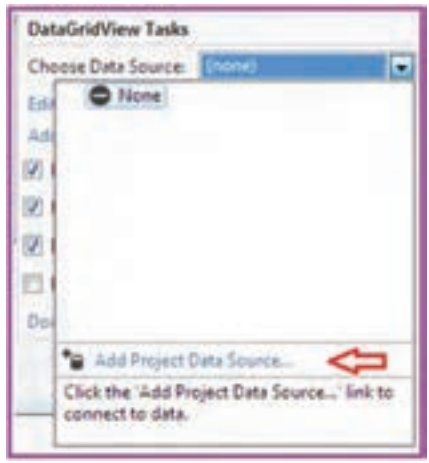
شکل ۲۴ــ۶ــ ب

۲۰ــ با اضافه کردن کنترل DataGridView، کادری در کنار آن اضافه می‌شود که در آن باید منبع اطلاعاتی این جدول نمایشی را معین کنید. همان طور که در ابتدای این فصل بیان شد یک جدول نمایشی نمی‌تواند مستقیماً به پایگاه داده متصل شود بلکه باید از طریق واسطه‌های مختلف که توضیح داده شد، عمل اتصال به داده‌ها انجام شود (شکل ۲۵ــ۶).



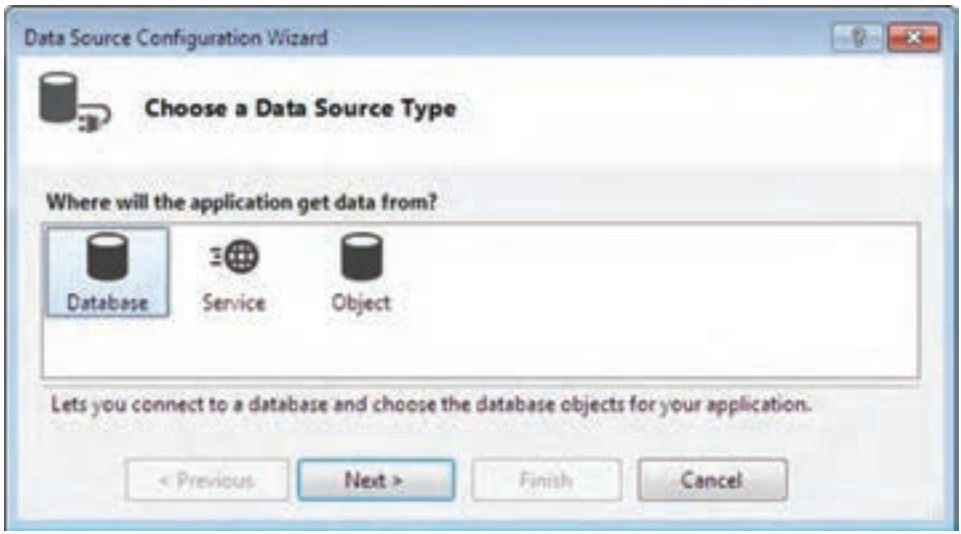
شکل ۲۵ــ۶

۲۱ــ برای ایجاد واسطه‌های مورد نیاز، بر روی گزینه Choose Data Source کلیک کنید و سپس بر روی گزینه ...Add Project Data Source کلیک کنید (شکل ۲۶ــ۶).



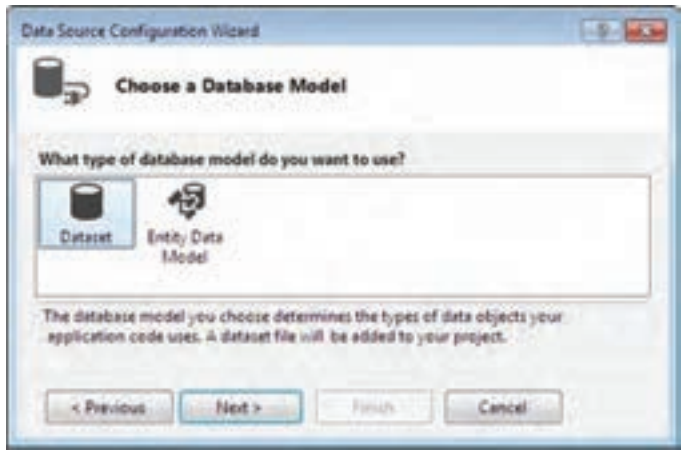
شکل ۲۶ــ۶ــ اضافه‌کردن منبع داده به پروژه

۲۲ــ چند مرحله را باید به ترتیب دنبال کنیم تا واسطه‌ها ساخته شوند. ابتدا گزینه Database را به‌عنوان نوع منبع اطلاعات در شکل ۲۷ــ۶ مشخص می‌کنیم.



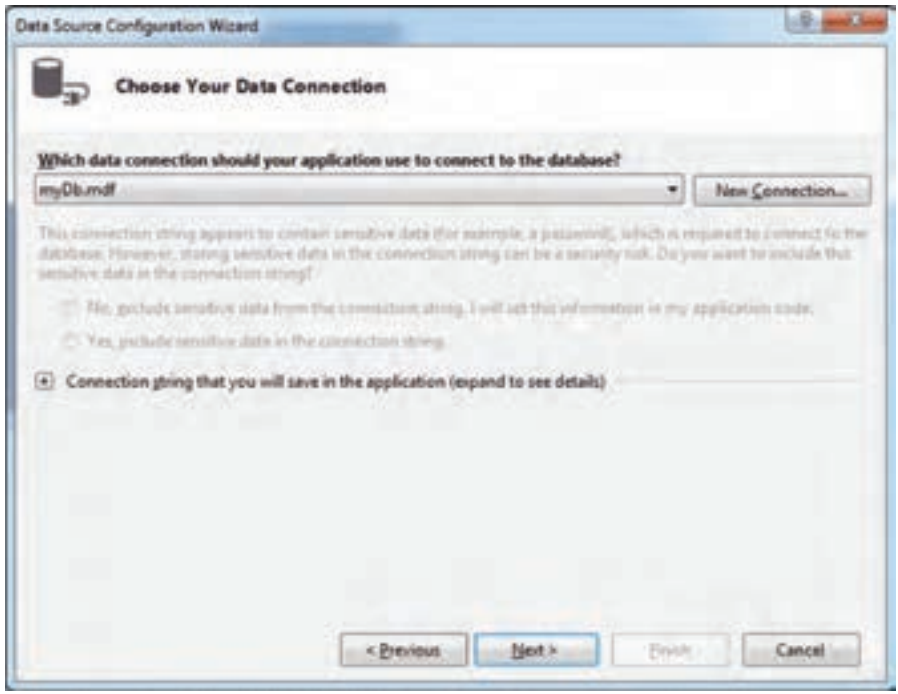
شکل ۲۷ــ۶ــ انتخاب نوع منبع داده

۲۳ــ بر روی دکمه Next کلیک کنید. در شکل ۲۸ــ۶، گزینه Dataset را به‌عنوان نمایش‌دهنده پایگاه داده در حافظه تعیین می‌کنیم.



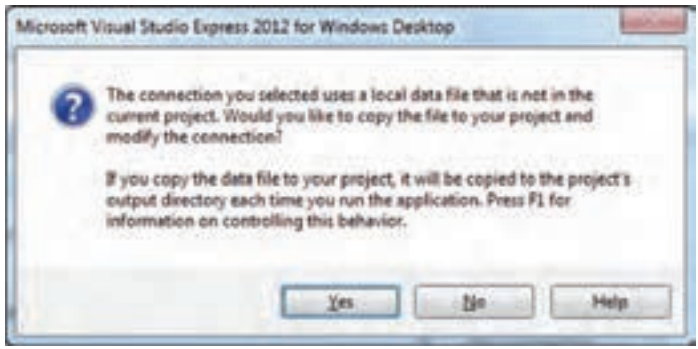
شکل ۲۸ــ۶ــ انتخاب مدل استفاده از پایگاه داده

۲۴ــ به مرحلهٔ بعدی می‌رویم و نام اتصال به پایگاه داده را مشخص می‌کنیم (شکل ۲۹ــ۶).



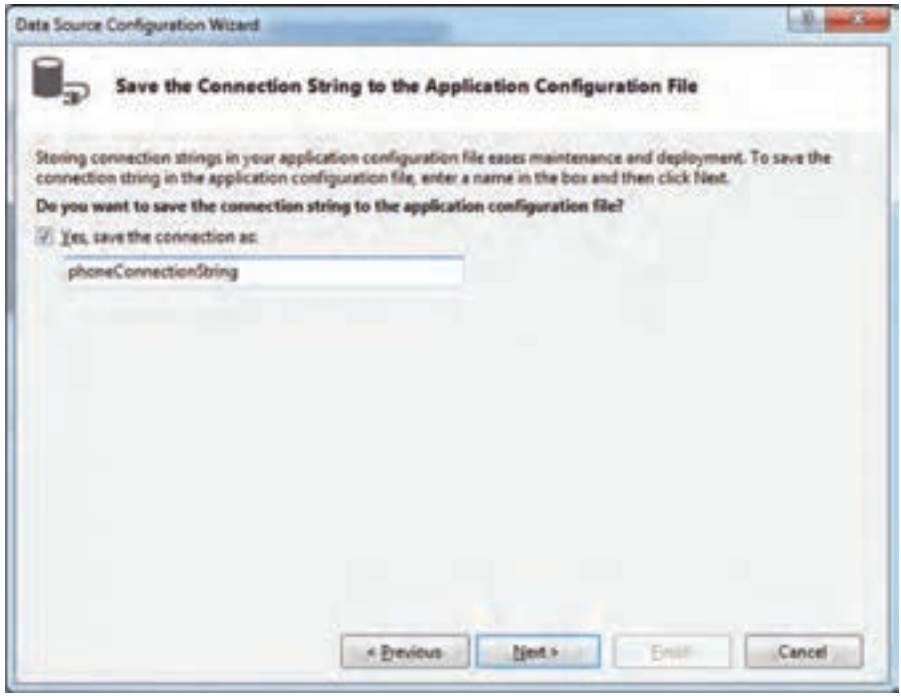
شکل ۲۹ــ۶ــ انتخاب اتصال

با زدن دکمه Next کادری نمایش داده می‌شود. دکمه Yes را بزنید. با این کار فایل پایگاه داده در پروژه کپی می‌شود و جزئی از پروژه محسوب می‌شود. بنابراین به‌صورت خودکار با ساختن فایل خروجی، در مسیر پوشه خروجی، فایل پایگاه داده نیز کپی می‌شود (شکل ۳۰ــ۶).



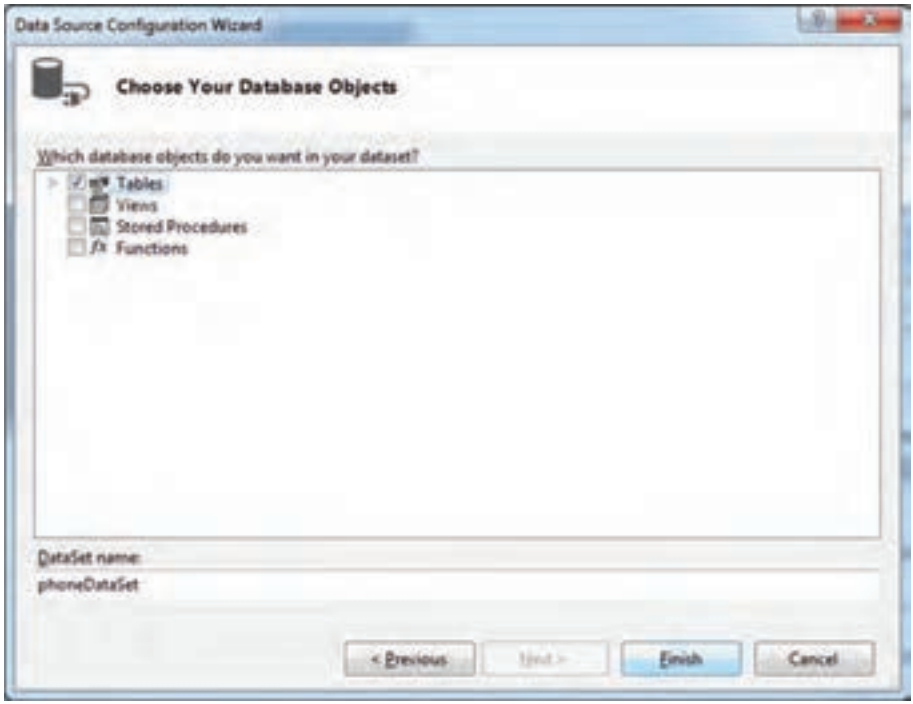
شکل ۳۰ ــ۶

۲۵ــ در مرحله بعدی تنظیم‌های انجام شده در برنامه ذخیره می‌شود. می‌توانیم نام اتصال را به‌صورت دلخواه تعیین نماییم. نام را phoneConnectionString انتخاب می‌کنیم (شکل ۳۱ــ۶).



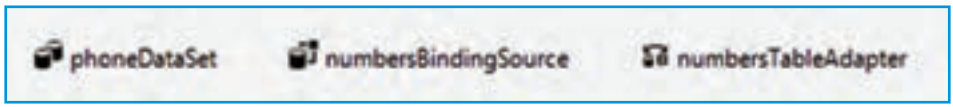
شکل ۳۱ــ۶ــ تعیین نام اتصال

۲۶ــ در مرحله بعد، جدول یا جدول‌هایی را که می‌خواهیم از طریق Dataset به آن دسترسی داشته باشیم (تمام جداول (Tables)) با فعال کردن کادر تیک مشخص می‌کنیم. همچنین نام dataset را نیز می‌توانیم به صورت دلخواه phoneDataSet تعیین کنیم. پس از انجام کارهای لازم، دکمه Finish را به منزله پایان عملیات کلیک کنید (شکل ۳۲ــ۶).



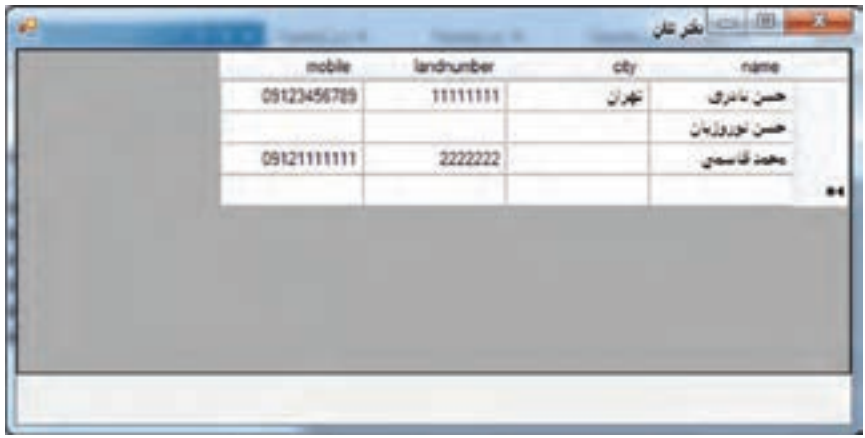
شکل ۳۲ــ۶ــ انتخاب جدول‌های موجود در پایگاه داده برای ورود در Dataset

۲۷ــ به این ترتیب تمام واسطه‌های مورد نیاز به طور خودکار و بدون برنامه‌نویسی ساخته می‌شود. اگر قسمت پایین صفحه در پنجره طراحی فرم را مشاهده کنید، واسطه‌های مورد نیاز ساخته شده‌اند (شکل ۳۳ــ۶). phoneDataSet نمایش پایگاه داده در حافظه RAM کامپیوتر است. numbersBindingSource واسطه بین DataGridView و Dataset است و numbersTableAdapter نیز واسطه بین پایگاه داده فیزیکی (ذخیره شده بر روی هارد دیسک) و نمایش پایگاه در حافظه (Dataset) است.



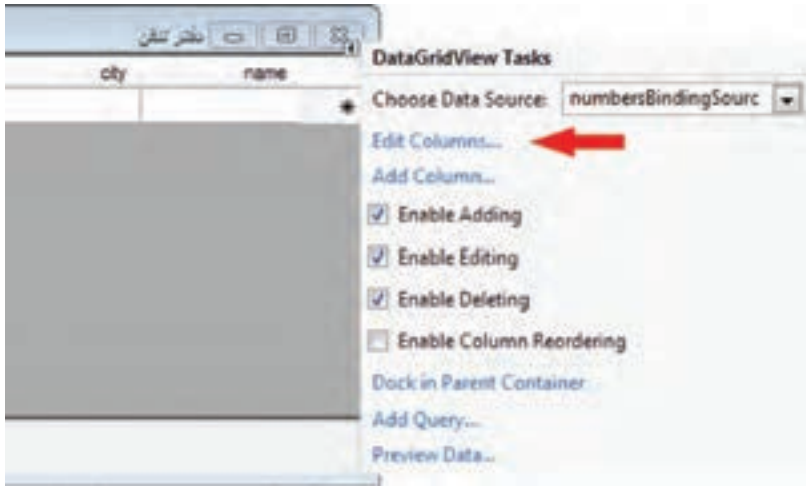
شکل ۳۳ــ۶ــ واسطه‌های اضافه‌شده به فرم

۲۸ــ اکنون می‌توانید برنامه را اجرا کنید. همان طور که در شکل ۳۴ــ۶ مشاهده می‌کنید، اطلاعات وارد شده در جدول numbers پایگاه داده در داخل جدول نمایشی بر روی فرم دیده می‌شود.



شکل ۳۴ــ۶

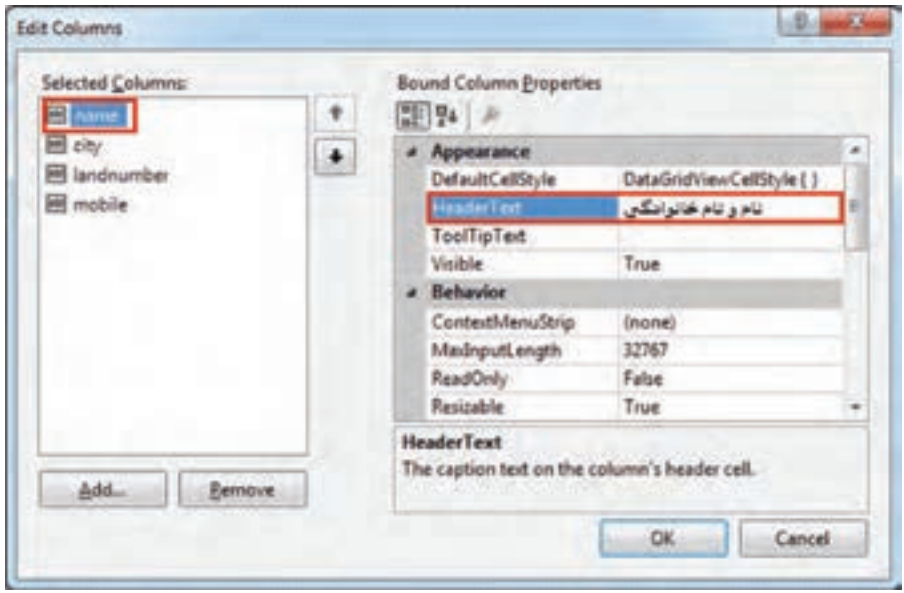
۲۹ــ همانطور که در شکل ۳۴ــ۶ ملاحظه می‌کنید، آنچه که در نمایش دفتر تلفن چندان دلچسب نیست عناوین ستون‌ها است که به زبان انگلیسی است و همچنین جهت نمایش اطلاعات است که از چپ به راست است و در زبان فارسی چندان مناسب نیست. برای تغییر در ستون‌ها، با استفاده از گزینه Edit Columns در منوی شکل ۳۵ــ۶ می‌توانید ستون‌ها را مطابق با میل خود تغییر داده و به اصطلاح سفارشی[1] کنید.



شکل ۳۵ــ۶ــ فرمان نمایش کادر ویرایش ستون‌ها

۱ــ Customize

۳۰ــ در شکل ۳۶ــ۶ ملاحظه می‌کنید که چگونه می‌توانید عنوان ستون‌ها را فارسی کنید.



شکل ۳۶ــ۶ــ تغییر متن عنوان ستون

آنچنان که در شکل ۳۶ــ۶ نمایش داده شد عنوان ستون name تغییر می‌کند. این عملیات را برای سایر ستون‌ها نیز انجام دهید.

همچنین می‌توانید در صورت لزوم، عرض ستون را توسط ویژگی Width تغییر دهید.

۳۱ــ اکنون برنامه را اجرا کنید و دفتر تلفن را مشاهده کنید. اگر بر روی عنوان ستون‌ها کلیک کنید، می‌توانید لیست را براساس آن مرتب شده مشاهده کنید.

۳۲ــ به سطر آخر جدول که با علامت * مشخص شده است مراجعه کرده و اطلاعات یکی از دوستان خود را در آن وارد کنید. در واقع یک رکورد جدید ایجاد کنید. سپس بین رکوردها حرکت کنید، به اولین رکورد بروید و به آخرین رکورد بازگردید.

۳۳ــ اکنون از برنامه خارج شوید و دوباره برنامه را اجرا کنید. به دفتر تلفن توجه کنید. آیا رکوردی که اضافه کردید، وجود دارد؟ چرا در پایگاه داده ذخیره نشده است؟

۳۴ــ هر گاه اطلاعات درون یک Dataset را تغییر دادید، برای اعمال تغییرات بر روی پایگاه داده فیزیکی، باید متد Update شیء TableAdapter را فراخوانی کنید.

۳۵ــ بدین منظور یک دکمه در جای مناسبی از فرم قرار دهید و ویژگی‌های آن را به‌صورت جدول ۶ــ۶ تنظیم کنید :

جدول ۶ــ۶ــ ویژگی‌های دکمه

دکمه Button	
مقدار	ویژگی
save	Name
ثبت تغییرات	Text

۳۶ــ در متد EH مربوط به رویداد کلیک دکمه ، دستور زیر را بنویسید.

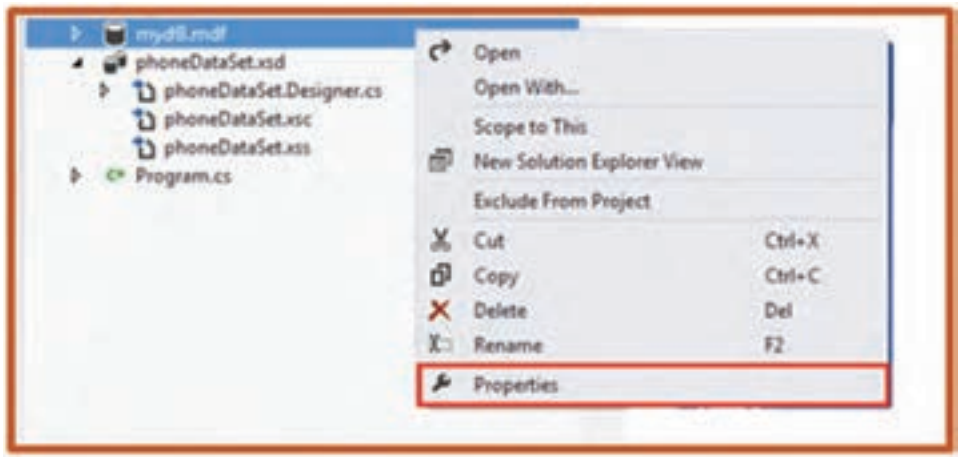
این کد تغییرات داده شده در جدول numbers از جدول phoneDataSet را به جدول numbers در پایگاه داده منتقل می‌کند.

```
private void save_Click(object sender, EventArgs e)
{
  numbersTableAdapter.Update(phoneDataSet.numbers);
}
```

۳۷ــ پروژه را اجرا کنید و رکوردهای جدیدی به دفترچه تلفن اضافه کنید و آن را ذخیره نمایید. اکنون برنامه را ببندید و دوباره پروژه را اجرا نمایید. آنچه مشاهده خواهید کرد همان رکوردهای اولیه می‌باشد. دلیل این اتفاق آن است که فایل پایگاه داده با اجرا و Debug توسط ویژوال استودیو فایل پایگاه داده، در محل فایل اجرایی کپی می‌شود و در واقع نرم‌افزار ایجاد شده، با این فایل پایگاه داده کار می‌کند. انجام عمل کپی فایل پایگاه داده به محل فایل اجرایی (پوشه Debug یا Release) در هر بار اجرا صورت می‌گیرد، بنابراین فایل قبلی که حاوی آخرین داده‌ها بوده است حذف شده و فایل قبلی جایگزین می‌شود.

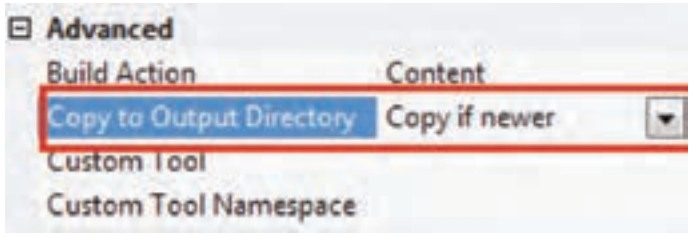
برای آنکه عملیات کپی فقط درصورتی انجام شود که فایل پایگاه داده ساخته نشده باشد یا از فایل ابتدایی قدیمی‌تر باشد به‌صورت زیر عمل کنید :

روی فایل پایگاه داده در پنجره Solution Explorer کلیک راست کنید و گزینه properties را بزنید (شکل ۳۷ــ۶).



شکل ۳۷ــ۶ــ گزینه Properties برای پایگاه داده

سپس مقدار گزینه Copy to Output Directory را به Copy if newer تغییر دهید (شکل ۳۸ــ۶).



شکل ۳۸ــ۶ــ تنظیم ویژگی Copy to output Directory

نکته

در برنامه دفتر تلفن، در پنجره Solution Explorer روی Form1 کلیک راست و پنجره کدنویسی را فعال کنید. به متد ()Form1_Load توجه کنید. دستور فراخوانی متد Fill شیء numbersTableAdapter را مشاهده می‌کنید. به وسیله فراخوانی متد ()Fill است که

اطلاعات درون پایگاه داده به داخل حافظه و در شیء DataSet قرار می‌گیرد. (واژه Fill به چه معنا است؟ آیا با عملکرد آن متناسب است؟)

```
private void Forml_Load (object sender, EventArgs e)
{
    // TODO:  This line of code loads data into the phoneDataSet.
    numbers table.
    //              You can move, or remove it, as needed.
    this.numbersTableAdapter. Fill (this.phoneDataSet.numbers);
}
```

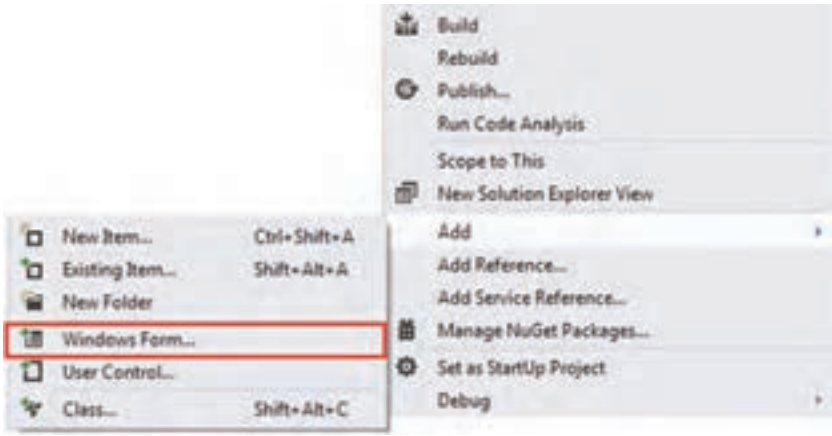
کار در کارگاه ۲

مثال ۲ــ۶ : می‌خواهیم فرمی‌داشته باشیم که اطلاعات وارد شده در دفتر تلفن را برای نام مورد جستجوی ما نمایش دهد.

الگوریتم و روش انجام کار : فرم جستجو را ایجاد می‌کنیم تا با زدن یک دکمه در فرم اصلی، نمایش داده شود. در فرم جستجو، نام شخص در کادر متنی نوشته می‌شود و با زدن دکمه جستجو، اگر رکوردی برای آن فرد ثبت شده باشد اطلاعات تماس در چند برچسب نمایش داده می‌شود (شکل ۳۹ــ۶).

شکل ۳۹ــ۶ــ فرم جستجو در پروژهٔ دفتر تلفن

۱ــ برای افزودن فرم جدید باید ابتدا کلاس فرم جدیدی تعریف کرد. این کار با استفاده از امکانات ویژوال استودیو بسیار آسان است. روی نام پروژه کلیک راست نمایید تا منوی آن نمایش داده شود. گزینه Windows Form را از منوی Add کلیک نمایید (شکل ۴۰ــ۶).



شکل ۴۰ــ۶ــ افزودن فرم جدید

۲ــ با کلیک این گزینه، کادر افزودن بخش جدید باز می‌شود که به‌صورت خودکار گزینه Windows Form انتخاب شده است.

۳ــ با واردکردن نام فرم جدید و زدن دکمه Add فرم جدید ایجاد خواهد شد[۱].

۴ــ در پروژه دفتر تلفن، فرمی با مشخصات جدول ۷ــ۶ اضافه می‌کنیم (شکل ۴۰ــ۶).

جدول ۷ــ۶ــ ویژگی‌های فرم

فرم Form		
ویژگی		مقدار
Name		Form2
Text		جستجو
FormBorderStyle		FixedToolWindow
Size	Width	324
	Height	317

۱ــ مراحل افزودن جدید در فعالیت پروژه فصل ۴ آورده شده است.

۵ــ در این فرم یک کادر متنی، یک دکمه و سه برچسب ایجاد می‌کنیم و در جای مناسب قرار می‌دهیم :

جدول ۸ــ۶ــ ویژگی‌های دکمه

دکمه Button	
مقدار	ویژگی
search	Name
جستجو	Text

جدول ۹ــ۶ــ ویژگی‌های کادر متنی

کادر متنی TextBox	
مقدار	ویژگی
nameBox	Name
	Text
Yes	RightToLeft

جدول ۱۰ــ۶ــ ویژگی‌های برچسب‌ها

برچسب Label				
ویژگی		مقدار	مقدار	مقدار
Name		cityLb	landNumberLb	mobileLb
RightToLedt		Yes	Yes	Yes
AutoSize		False	False	False
BackColor		white	white	white
TextAlign		MiddleCenter	MiddleCenter	MiddleCenter
Size	Width	206	206	206
	Height	33	33	33

همچنین چهار برچسب دیگر برای برچسب‌های نام، شهر، شماره منزل و شماره همراه ایجاد کرده و در جای مناسب قرار می‌دهیم.

۶ــ در فرم اصلی (Form1) نیز یک دکمه ایجاد می‌کنیم تا با زدن آن، فرم جستجو باز شود (شکل ۴۱ــ۶).

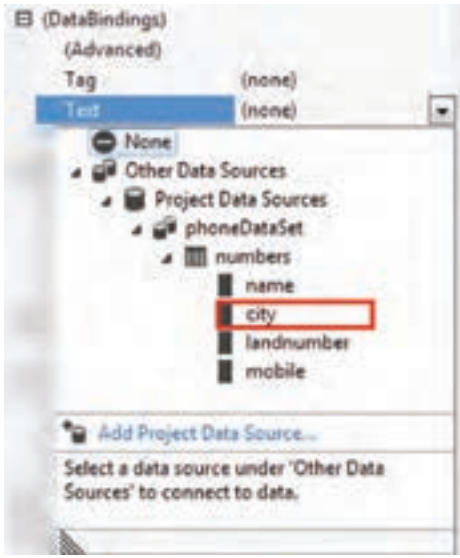
جدول ۱۱ــ۶ــ ویژگی دکمه

دکمه Button	
مقدار	ویژگی
showSearchForm	Name
جستجوی اطلاعات	Text

شکل ۴۱ــ۶ــ دکمه جستجوی اطلاعات در فرم اصلی

اتصال کنترل‌ها به پایگاه داده

۷ــ اکنون طراحی واسط کاربری به پایان رسیده است و نوبت اتصال به پایگاه داده است. برچسب cityLbl را انتخاب نمایید. در پنجره properties و در گروه Data با زدن علامت + گزینه (DataBindings) آن را باز نمایید. سپس مطابق شکل ۴۲ــ۶ با کلیک روی علامت مثلث شکل، کادر تنظیم آن را باز کنید و برای گزینه Text، مسیر را مطابق شکل طی کرده و فیلد city را انتخاب نمایید.



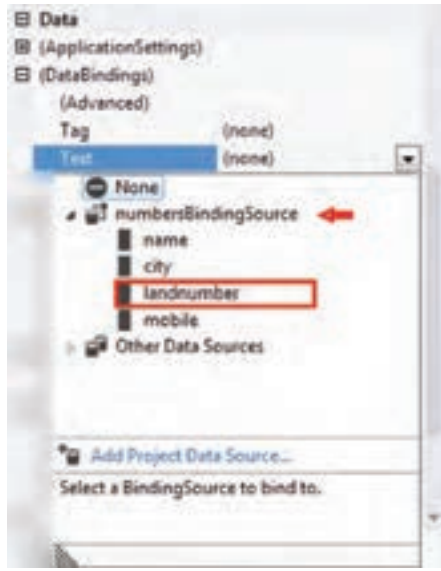
شکل ۴۲ ــ۶

با این عملیات ما از phoneDataSet موجود در پروژه استفاده می‌کنیم و ویژگی متن (Text) برچسب cityLbl را به فیلد city در جدول numbers مقید می‌کنیم. با انجام این کار، یک شیء BindingSource و یک TableAdapter و همچنین خود DataSet به‌صورت خودکار اضافه می‌شود (شکل ۴۳ــ۶).

شکل ۴۳ــ۶ــ واسطه‌های اضافه‌شده به فرم جستجو

۸ــ اکنون چون یک BindingSource ایجاد شده است برای متصل کردن ویژگی متن برچسب شماره منزل (landNumberLbl) به‌صورت زیر عمل می‌کنیم :

در کادر تنظیم گزینه Text از بخش (Databinddings)، گزینه numbersBindingSourca را باز کنید و فیلد landnumber را انتخاب نمایید.



شکل ۴۴ــ۶

برای شماره همراه (mobileLbl) نیز به همین روش عمل کرده و آن را به فیلد mobile مقید کنید.

کد نویسی :

۹ــ اکنون کد متد EH رویداد کلیک دکمه showSearchForm در فرم اصلی (Form1) را

به صورت زیر می‌نویسیم :

```
private void showSearchForm_Click(object sender, EventArgs e)
{
    Form2 f 2 = new Form2();
    f 2 . ShowDialog();
}
```

۱۰ــ برنامه را اجرا کنید و با دکمه «جستجوی اطلاعات» در فرم اصلی حاصل کار را بررسی کنید.

شکل ۴۵ــ۶

همان طور که می‌بینید مشخصات اولین رکورد در برچسب‌ها آمده است. این موضوع به دلیل آن است که ما برچسب‌ها را به طریقی که پیش‌تر آمد به فیلدها مقید کرده‌ایم. اما می‌خواهیم فقط با عمل جستجو، اطلاعات فرد نمایش داده شود. بنابراین لازم است که مقیدسازی این برچسب‌ها در ابتدای کار به حالت تعلیق در بیاید. برای این کار در متد سازنده Form2 کد زیر را اضافه می‌نماییم :

```
public Form2()
{
    InitializeComponent();
    numbersBindingSource . SuspendBinding();
}
```

متد SuspendBinding در کد بالا، مقیدسازی را برای numbersBindingSource به‌حالت تعلیق در می‌آورد. اگر اکنون برنامه را اجرا کنیم خواهید دید که متن برچسب‌ها خالی است.

۸ـ برای عمل جستجو، متد EH رویداد کلیک دکمه را به صورت زیر بنویسید.

```
private void search_Click(object sender, EventArgs e)
{
    int find = numbersBindingSource.Find("name", nameBox.Text);
    if (find >= 0)
    {
        numbersBindingSource.ResumeBinding();
        numbersBindingSource.Position = find;
    }
    else
        numbersBindingSource.SuspendBinding();
}
```

دستور numbersBindingSource.Find("name", nameBox.Text) عمل جستجو را در رکوردها می‌پردازد و شماره اولین رکوردی که فیلد name آن برابر با متن درون کادر متنی nameBox باشد برگشت می‌دهد. این شماره از صفر شروع می‌شود. یعنی اگر نام و نام‌خانوادگی اولین رکورد همان چیزی باشد که در کادر متنی nameBox وارده شده است، آن‌گاه مقدار صفر را برمی‌گرداند. این مقدار برگشتی را درون یک متغیر به نام find قرار می‌دهیم تا بعد از آن استفاده کنیم.

اکنون سؤال اینجاست که اگر رکوردی پیدا نشد که مطابق جستجوی ما باشد چه عددی برگشت داده می‌شود؟ جواب این است که در این صورت عدد برگشتی عدد ۱ – می‌باشد. از همین نکته می‌توان استفاده کرد تا عملیات مورد نظر را در صورت پیدا شدن رکورد یا پیدا نشدن آن، طرح ریزی کرد.

شرط if (find >= 0) بررسی می‌کند که اگر نتیجه جستجو، بزرگتر یا مساوی 0 باشد (رکوردی مطابق با جستجو پیدا شود) دستورهای مناسب را انجام دهد. این دستورها به صورت زیر می‌باشند :

```
numbersBindingSource.ResumeBinding();
numbersBindingSource.Position = find;
```

دستور اول حالت مقید سازی داده‌ها را از سر می‌گیرد و به عبارتی آن را دوباره فعال می‌کند. چرا که پیش از این ممکن است در حالت تعلیقی بوده باشد. دستور دوم، موقعیت رکورد جاری در لیست رکوردها را مشخص می‌کند. در هنگام مقداردهی، اطلاعات رکورد جاری در کنترل‌های متصل استفاده می‌شود.

در انتها کد

```
else
    numbersBindingSource.SuspendBinding();
```

مشخص می‌کند که در صورت درست نبودن شرط یعنی پیدا نشدن هیچ رکوردی، مقیدسازی داده‌ها به حالت تعلیق درآید تا بدین ترتیب متن برچسب‌های مورد نظر، خالی شود.

اکنون برنامه را اجرا کنید و با وارد کردن نام شخص، حاصل کار را بررسی کنید (شکل ۴۶ــ۶).

شکل ۴۶ ــ۶

خودآزمایی فصل ششم

الف) درستی یا نادرستی هر عبارت را تعیین کنید.

۱ــ اطلاعات درون یک پایگاه داده را در برنامه‌های ویژوال استودیو نمی‌توان ویرایش کرد.

۲ــ در جستجو توسط متد find از واسط BindingSource، اگر رکوردی پیدا نشود مقدار صفر برگردانده می‌شود.

ب) جاهای خالی را با عبارت مناسب پر کنید.

۳ــ هر یک از اجزای تشکیل دهنده یک رکورد را ..... می‌نامند.

۴ــ یک کپی از داده‌های پایگاه داده در حافظه RAM کامپیوتر، به وسیله شیء ..... ایجاد می‌شود.

۵ــ ورود اطلاعات درون پایگاه داده به داخل حافظه و در شیء DataSet، به وسیله فراخوانی متد ..... انجام می‌گیرد.

۶ــ برای از سر گرفتن مقید سازی داده‌ها از متد .....استفاده می‌شود.

۷ــ برای تعیین موقعیت رکورد جاری در لیست رکوردها در واسط BindingSource از متد ..... استفاده می‌شود.

ج) به سؤالات زیر پاسخ دهید.

۸ــ مزیت‌های استفاده از پایگاه داده به جای فایل برای ذخیره داده‌ها در حافظه جانبی را بنویسید.

۹ــ جدول زیر را تکمیل نمایید.

ابزار پایگاه داده	کاربرد
DataSet	
DataGridView	
TableAdapter	
BindingSource	

۱۰ــ به چه دلیل در تنظیمات پایگاه داده مقدار ویژگی Copy to Output Directory را به Copy if newer تغییر می‌دهید؟

۱۱ــ در ارتباط با پایگاه داده به چه دلیل متد SuspendBinding فراخوانی می‌شود؟

تمرینات برنامه‌نویسی فصل ششم

۱ــ یک برنامه برای نگهداری اطلاعات گوشی‌های موبایل ایجاد کنید. این برنامه از DataGridView برای نمایش و ثبت و ویرایش در جدول پایگاه داده استفاده می‌کند. فیلدهای اطلاعاتی که در مورد هر گوشی اهمیت دارد و باید نگهداری شود را پیدا کنید و در جدول به‌کار بگیرید.

۲ــ پروژه‌ای ایجاد نمایید که شامل پایگاه داده‌ای برای نگهداری مشخصات دانش‌آموزان شامل کد دانش‌آموزی، نام کامل (نام و نام خانوادگی) و سه نمره درسی باشد. توسط DataGrid اطلاعات دانش آموزان را وارد نمایید. در فرم جستجو کد دانش‌آموز وارد شود و در صورت پیداشدن دانش‌آموز، نام و میانگین نمرات وی نمایش داده شود.

فعالیت

پایان پروژه

در بخش پایانی پروژه، ضمن عرض خدا قوت به شما هنرجویان و هنرآموزان عزیز، پایگاه داده‌ها را برای واژگان تشکیل می‌دهیم.

مرحله ششم:

در مرحله پایانی تکمیل پروژه، راهکار استفاده از پایگاه داده برای نگهداری اطلاعات واژگان را تمرین می‌کنیم.

۱ــ آرایه‌های words و meaning را حذف کنید. همچنین عملیات مربوط به جستجو نیز باید حذف شود.

۲ــ یک فایل پایگاه داده ایجاد کنید.

۳ــ جدولی به نام words به صورت زیر درون پایگاه داده جدید ایجاد نمایید.

Name	Data Type	Allow Nulls
word	nvarchar(150)	☐
description	nvarchar(1000)	☐
		☐

۴ــ یک کنترل DataGridView در فرم RegisterAndEditForm اضافه کنید و آن را به جدول words مقید کنید.

۵ــ یک دکمه به نام save و با متن «ذخیره» به فرم RegisterAndEditForm اضافه نمایید. همچنین ستون‌های آن را به «واژه» و «معنی و توضیح» تغییر دهید.

فرم register And Edit در زمان اجرا

۶ــ در فرم اصلی، کادر معنی و توضیح (description) را به فیلد description در جدول words مقید کنید. در حین عمل اشیاء BindingSource، DataSet و TableAdapter به صورت زیر ساخته می‌شود.

۷ــ برای اینکه مقید شدن کنترل در ابتدا به تعلیق در آید و مقداری در کادر متنی description نمایش داده نشود کد زیر را به متد سازنده Form1 اضافه کنید.

```
public form1( )
{
    InitializeComponent ( );
    wordBindingSource. SuspendBinding ( );
}
```

کدی که در متد EH رویداد کلیک دکمه جستجو نوشته اید را حذف کنید و کد زیر را جایگزین کنید :

```
private void search_Click (object sender, EventArgs e)
{
    Int find = wordsBindingSource. Find ("word",searchword.Text);
    if (find > 0)
    {
    wordsBindingSource. ResumeBinding ( );
    wordsBindingSource. position = find;
}
else
{
    wordsBindingSource.SuspendBinding ( );
    MessageBox.Show ("پیدا نشد");
  }
}
```

در این کد ابتدا نتیجه جستجو توسط متد Find برای واسط wordsBindingSource در متغیر find ذخیره می‌شود. سپس این نتیجه مورد بررسی قرار می‌گیرد. اگر رکوردی پیدا شده باشد، مقیدسازی ادامه پیدا می‌کند و کادر متنی description، فیلد توضیح (description) مربوط به رکورد پیدا شده را نمایش می‌دهد. در غیر این صورت مقیدسازی به تعلیق در می‌آید و پیام متناسب نمایش داده می‌شود.

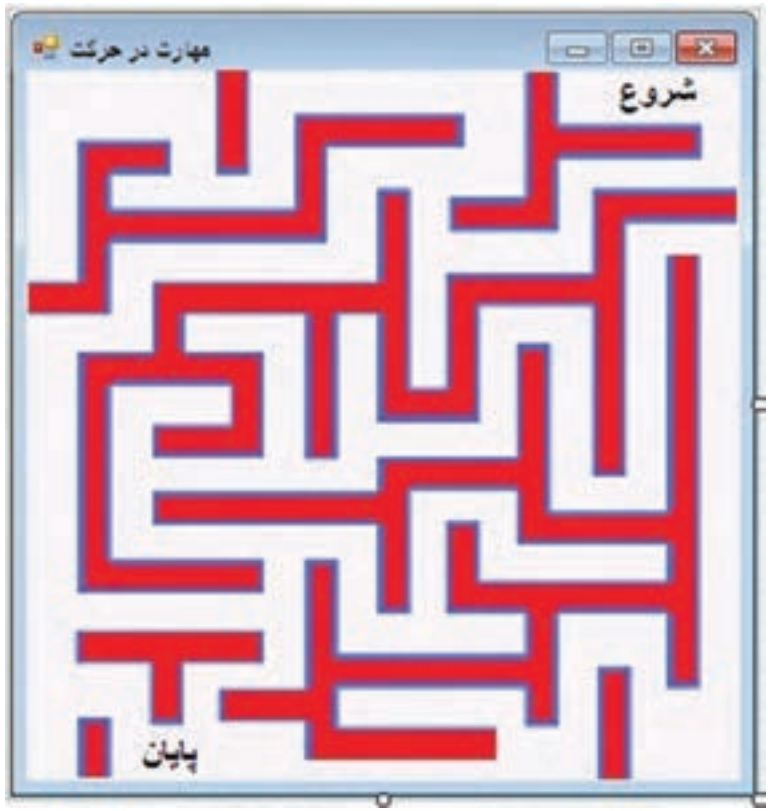
واژگان و اصطلاحات انگلیسی فصل ششم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Adapter	
۲	Allow	
۳	Binding	
۴	Compact	
۵	Connection	
۶	Continue	
۷	Data base	
۸	Data type	
۹	Edition	
۱۰	Express	
۱۱	Fill	
۱۲	Find	
۱۳	Grid	
۱۴	Primary	
۱۵	Set	
۱۶	Source	
۱۷	Table	
۱۸	Update	
۱۹	View	

پیوست ۱

تمرین برای هنرجوی سخت کوش (فصل دوم)

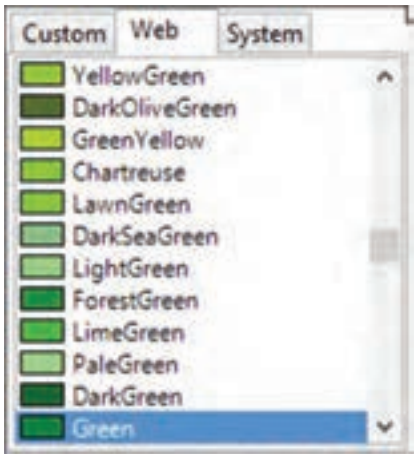
* بازی مهارت در حرکت ماوس یا تست را بسازید.



تعدادی برچسب مطابق با شکل بالا در روی فرم قرار دهید و رویداد MouseEnter را برای آنها فعال کنید. به طوری که به محض وارد شدن ماوس در هر یک از آنها پیام «شما باختید» را ظاهر نماید.

با توجه به اینکه تمام برچسب‌ها دارای یک متد EH یکسان برای رویداد MouseEnter می‌باشد، برای نوشتن متد EH برچسب‌ها، همه آنها را با یکدیگر انتخاب کنید و سپس در پنجره Properties بر روی رویداد MouseEnter دو بار کلیک کنید تا پنجره کدنویسی باز شود. سپس اقدام به نوشتن دستور MessageBox نمایید.

پیوست ۲



رنگ‌ها : توجه کنید که وقتی می‌خواهید رنگ پس زمینه یا متن کنترل را تغییر دهید در قسمت BackColor و یا ForeColor یک پالت رنگ باز می‌شود.

در این پالت سه زبانه وجود دارد :

۱ــ System : رنگ‌های سیستمی قابل انتخاب می‌باشند. این رنگ‌ها بستگی به سیستم دارند. مثلاً اگر Window را انتخاب کنیم و در سیستمی که رنگ داخل پنجره‌های آن آبی تعیین شده باشد رنگ نمایش داده شده در آن سیستم، آبی خواهد بود.

۲ــ Web : رنگ‌های مرسوم و استاندارد وب را عرضه می‌کند.

۳ــ Custom : از این زبانه می‌توان رنگ‌هایی را انتخاب کرد یا با تولید یک رنگ از آن استفاده نمود. برای تولید یک رنگ باید روی یک سلول کلیک راست نمود (مطابق شکل زیر). سپس از کادری که باز می شود رنگ مورد نظر را انتخاب و اضافه کرد.

بخشی از جدول نام‌های استاندارد و کد رنگ‌ها در web بر اساس کد مبنای ۱۶

پیوست ۳

تمرین برای هنرجوی سخت‌کوش (فصل چهارم)

مثال : می‌خواهیم برنامه‌ای بنویسیم که با به‌کار بردن کلاس و آرایه، یک دفتر تلفن ایجاد کند و در آن نام و شماره تلفن دوستان خود را وارد کنیم.

الگوریتم یا روش انجام کار : در این مثال، دفتر تلفن را به عنوان یک شیء در نظر می‌گیریم و سعی می‌کنیم ویژگی این مثال‌ها و عملیاتی که بر روی آن انجام می‌شود را شناسایی کنیم.

ویژگی‌های یک دفتر تلفن می‌تواند چنین باشد :

هر دفتر تلفن حداقل دارای دو قسمت برای نوشتن اطلاعات نام شخص و شماره تلفن است و یک ظرفیت محدود (تعداد صفحات) برای وارد کردن اسامی و تلفن‌ها را دارد.

بنابراین به دو فیلد نیاز داریم . فیلد name و فیلد phoneNumber

همچنین تعداد اسامی‌یا ردیف‌های وارد شده را نگهداری می‌کنیم تا از خالی بودن دفتر تلفن و یا پر شدن آن مطلع شویم. بنابراین فیلد سوم که برای کنترل است استفاده می‌کنیم و نام آن را numEntries انتخاب می‌کنیم.

یک متد برای اضافه کردن مشخصات یک فرد به دفتر تلفن نیاز داریم که ورودی آن ، نام و شماره تلفن فرد است.

یک متد نیز برای جستجوی تلفن یک شخص نیاز داریم. نام شخص به عنوان ورودی به این متد ارسال می‌شودو شماره تلفن فرد نیز به عنوان خروجی متد است.

بنابراین الگو و شکل کلاس چنین خواهد بود :

```
public class PhoneBook
{
    int numEntries;
    string [] names;
    string [] phoneNumbers;
public PhoneBook ( )
{
    numEntries = 0;
    names = new string [100];
    phoneNumbers = new string [100];
}
public void add (string name, string phoneNumber)
{
    names {numEntries] = name;
    phoneNumbers [numEntries] = phoneNumber;
    numEntries ++;
}
public string lookUp (string name)
{
    for (int cnt = 0; cnt , numEntries; cnt++)
  {
        if (name. Equals (names [cnt]) )
        {
            return phoneNumbers [cnt];
        }
    }
    return "";
}
}
```

پیوست ۴

متدهای پرکاربرد کلاس Directory

ردیف	نام متد	شرح
۱	CreateDirectory	یک فولدر جدید می‌سازد
۲	Delete	یک شاخه یا فولدر را پاک می‌کند
۳	Exists	اگر شاخه یا فولدر معینی از قبل وجود داشته باشد مقدار true در غیر این صورت مقدار flase را بر می‌گرداند.
۴	GetDirectories	آرایه‌ای از نوع رشته‌ای برمی‌گرداند، که شامل نام شاخه‌های موجود در یک فولدر خاص است.
۵	GetFiles	آرایه‌ای از نوع رشته‌ای برمی‌گرداند، که شامل نام فایل‌های موجود در یک فولدر مشخص است.
۶	Move	یک شاخه یا فولدر را به محل دیگری منتقل می‌کند.
۷	SetCurrentDirectory	مسیر جاری را تغییر می‌دهد.

متدهای پرکاربرد کلاس path

ردیف	نام متد	شرح
۱	GetDirectoryName	اطلاعات فولدر را براساس رشته ورودی برمی‌گرداند.
۲	GetFileName	اسم فایل که شامل پسوند فایل است را برمی‌گرداند.
۳	GetExtension	پسوند فایل را برمی‌گرداند.
۴	GetFileName WithoutExtension	اسم فایل بدون پسوند فایل را برمی‌گرداند.
۵	HasExtension	اگر مسیر شامل اسم فایل و پسوند فایل باشد true و در غیر این صورت false برمی‌گرداند.
۶	ChangeExtension	پسوند فایل را که به همراه مسیر ذکر شده است را تغییر می‌دهد.

جدول متدهای پرکاربرد کلاس file

ردیف	نام متد	شرح
۱	Create()	ایجاد یک فایل
۲	CreateText()	ایجاد یک فایل متنی
۳	Open()	باز کردن یک فایل
۴	OpenText()	باز کردن یک فایل متنی
۵	OpenRead()	باز کردن یک فایل فقط جهت خواندن اطلاعات درون آن
۶	OpenWrite()	بازکردن یک فایل فقط برای نوشتن و ذخیره کردن اطلاعات
۷	Delete()	یک فایل را حذف یا پاک می‌کند
۸	AppendText()	برای اضافه کردن متنی به انتهای یک فایل متنی موجود. اگر فایل از قبل وجود نداشته باشد آن را ایجاد می‌کند
۹	Copy()	یک کپی از فایل موجود تهیه می‌کند.
۱۰	Exists()	اگر فایل وجود داشته باشد مقدار خروجی متد برابر true است در غیر این صورت مقدار false بر می‌گرداند.
۱۱	Move()	یک فایل معین را به فولدر دیگری منتقل یا جابجا می‌کند
۱۲	WriteAllText()	می‌تواند یک فایل متنی جدید ایجاد کرده و متنی در آن بنویسید و سپس فایل را ببندد. اگر فایل وجود داشته باشد روی آن بازنویسی می‌کند
۱۳	AppendAllText()	می‌تواند یک فایل متنی را باز کرده و متنی را به انتهای آن اضافه کرده و سپس فایل را ببندد.
۱۴	ReadAllText()	یک فایل متنی موجود را باز کرده و محتوای آن را خوانده و به صورت یک رشته بر می‌گرداند. در پایان فایل را می‌بندد.