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؟

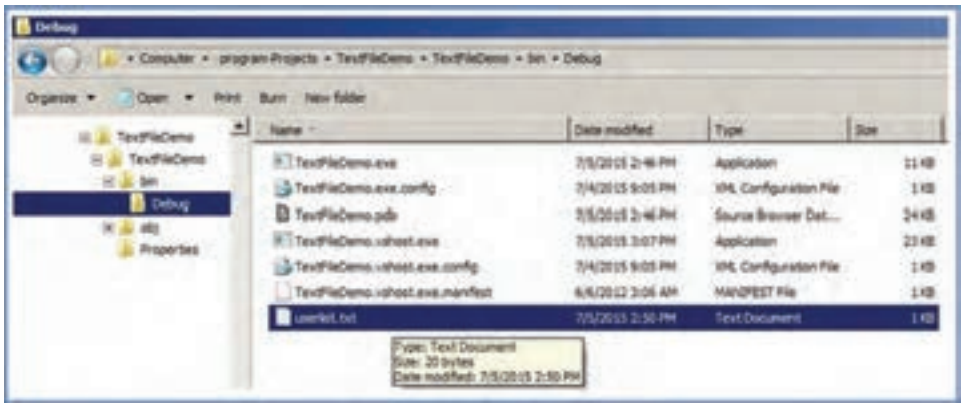
برنامه را اجرا کنید و اطلاعاتی را در فرم وارد کنید. اگر اطلاعات ناقص باشد خطایی متناسب با نقص اطلاعات مشاهده خواهید کرد (شکل ۲ــ۵).

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

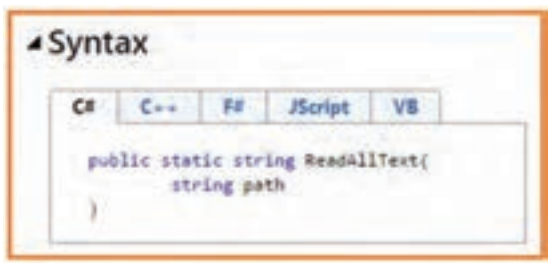
الگوریتم یا روش انجام کار : همانطور که برای خواندن یک کتاب، ابتدا کتابی را از قفسه برداشته، سپس آن را باز کرده، شروع به خواندن صفحه‌ای می‌کنیم و در انتها کتاب را بسته، در جای خود قرار می‌دهیم، برای خواندن فایل نیز باید عملیات زیر را انجام دهیم :

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

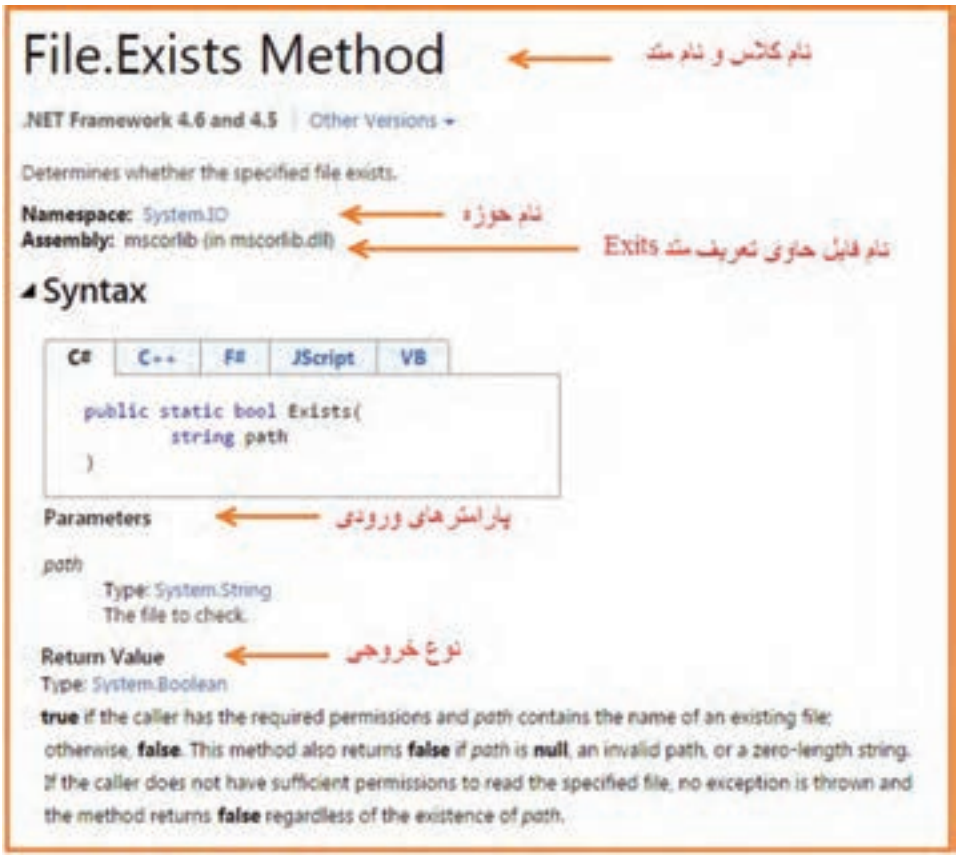
۲ــ اطمینان از وجود فایلی با نام وارد شده

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 :


File.ReadAllText Method (String)

.NET Framework 4.6 and 4.5 | Other Versions

Opens a text file, reads all lines of the file, and then closes the file.

Namespace: System.IO
Assembly: mscorlib (in mscorlib.dll)

Syntax

C# | C++ | F# | JScript | VB

```
public static string ReadAllText(
        string path
)
```

Parameters

path
Type: System.String
The file to open for reading.

Return Value
Type: System.String
A string containing all lines of the file.

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؟

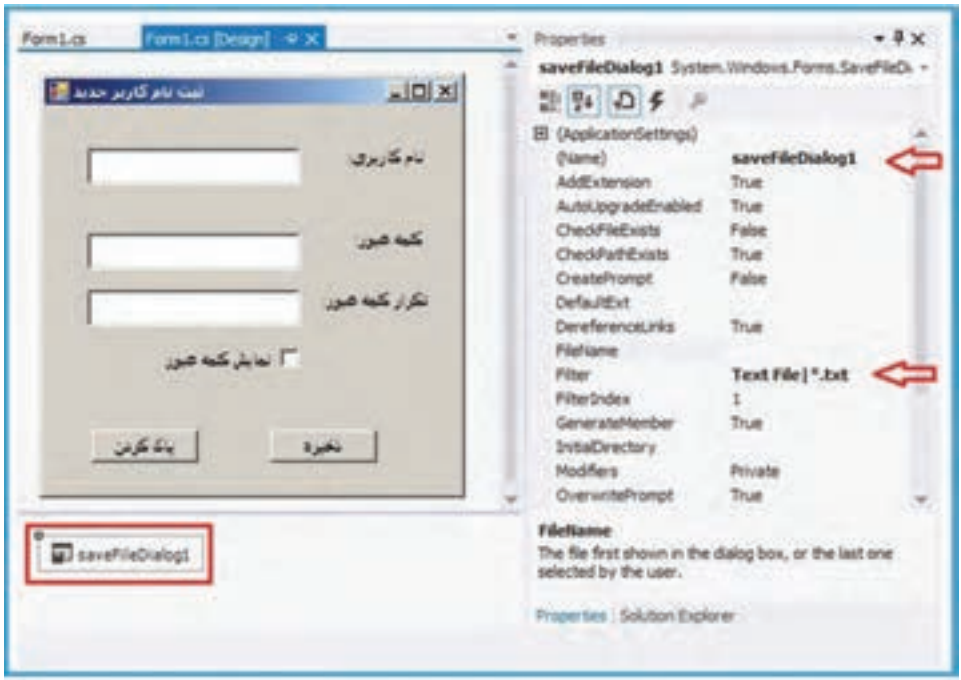
۱۳ــ دو روش دسترسی به اطلاعات درون فایل را با هم مقایسه کنید.

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.

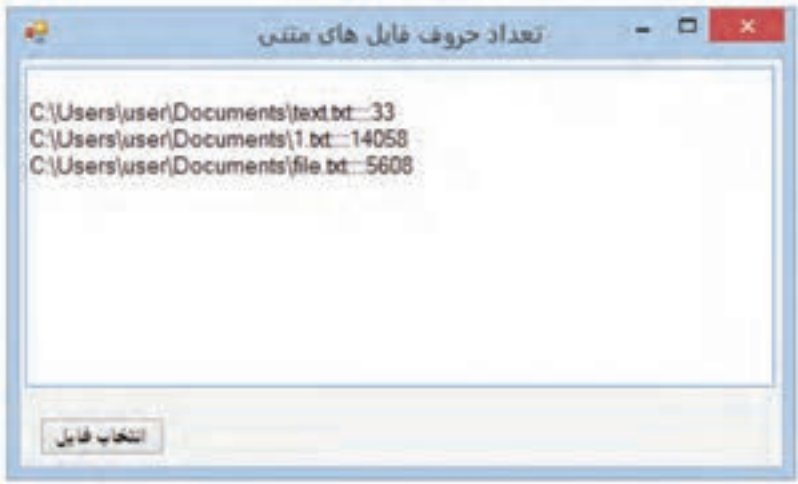
روش کار :

○ 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.



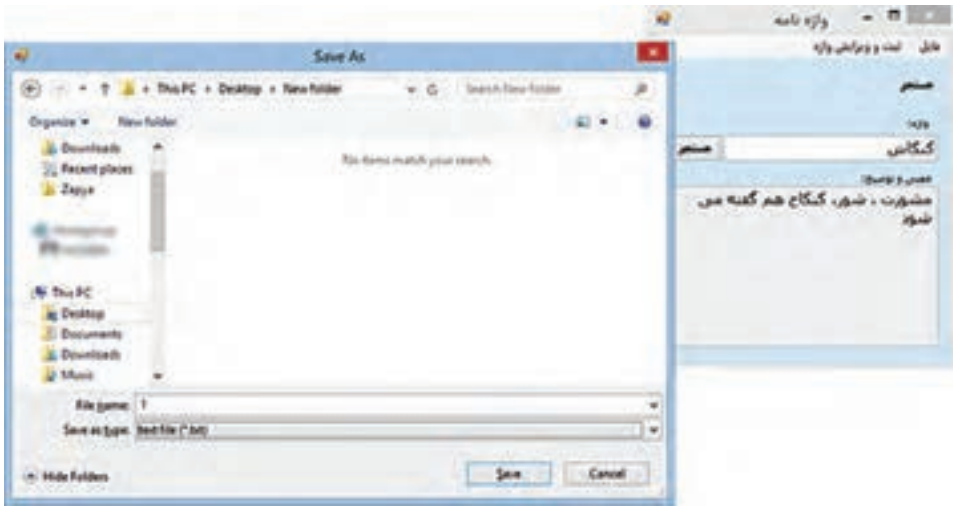
فعالیت

تکمیل پروژه

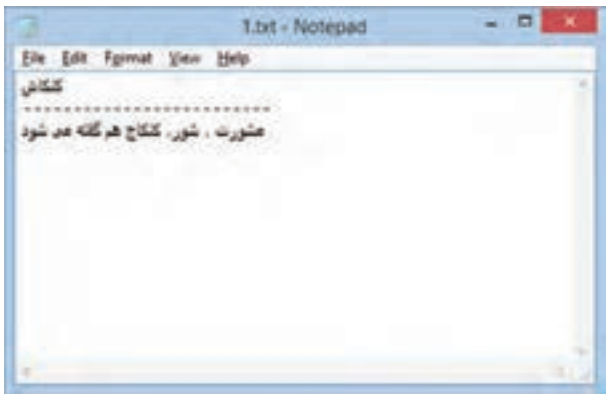
در این بخش از تکمیل پروژه، واژگان را در فایل ذخیره می‌کنیم.

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	