آموزش مقدماتی سی شارپ

مهدی ریزوندی

اگر شما اطلاعات ز یادی در خصوص برنامه نو یسی ندار ید و
می خواهید با زبان برنامه نویسی قدرتمند سی شارپ آشنایی
پیدا کنید، این نوشته برای شما ته یه شده است

ChargMagic

CSharpBlog.Blogfa.com

csharpblog@yahoo.com

1389/02/16

فهرست مطالب

داده‌ها در سی شارپ

bool: true یا false

byte: داده 8 بیتی بدون علامت

short: عدد صحیح 16 بیتی

int: عدد صحیح 32 بیتی

long: عدد صحیح 64 بیتی

float: عدد اعشاری 32 بیتی

double: عدد اعشاری 64 بیتی

Char: کاراکتر 16 بیتی

string: کاراکترهای 16 بیتی

شما می‌توانید یک داده سطح بالاتر را با داده سطح پایین تر مقداردهی کنید. (در اینجا منظور من از سطح مجموعه اعداد و حافظه متغیر می باشد)

```
float y = 7.0f; //y is of type float
int j;          //j is of type int
y = j;          //convert int to float
```

البته امکان مقداردهی یک داده سطح پایین تر با داده سطح بالاتر نیز وجود دارد. به این عمل اصطلاحا Casting گفته می‌شود.

```
1)
j = (int)y; //convert float to integer
2)
float x = 1.0E45;
int k = (int) x;
3)
int k;
bool gtnum;
gtnum = (k > 6); //true if k is greater than 6
```

در نظر داشته باشید که ممکن است عمل casting با خطا مواجه شود.

همچنین در نظر داشته باشید که امکان مقداردهی متغیرهای Boolean با اعداد همانند c یا c++ وجود ندارد. و تبدیل متغیر از نوع Boolean به دیگر متغیرها امکانپذیر نیست.

تبدیل انواع اعداد و رشته ها به یکدیگر

با استفاده از متد Convert می‌توانید اعداد را به رشته و رشته را به عدد تبدیل نمایید. البته متد Convert تبدیلات دیگری را نیز انجام می دهد که پس از تایپ دات بعد از متد می‌توانید همه آنها را ببینید.

```
string s = Convert.ToString(x);
float y = Convert.ToSingle(s);
float x = 12.341514325f;
string s = x.ToString("###.###"); //gives 12.342
```

تعیین نوع و مقدار برای متغیر در زمان تعریف:

```
float loan = 1.23f;      //float
long pig = 45L;          //long
int color = 0x12345;     //hexadecimal
```

در سی شارپ سه مقدار ثابت از قبل تعریف شده وجود دارد که عبارتند از:

```
true, false, null
```

برای قرار دادن کارکترهای خاص غیر رشته ای (عموما کارکترهای کنترلی) از رشته کاراکترهای از پیش تعریف شده استفاده می‌شود:

```
'\n' newline (line feed)
'\r' carriage return
'\t' tab character
'\b' backspace
'\f' form feed
'\0' null character
'\"' double quote
'\'' single quote
'\\' backslash
```

نامگذاری متغیر:

در نامگذاری متغیرهای دقت کنید که حروف کوچک با حروف بزرگ متفاوت هستند در نتیجه سه کلمه زیر سه متغیر مختلف را تعریف می‌کنند:

```
temperature
Temperature
TEMPERATURE
```

یکی از قابلیتهای سی شارپ در مقداردهی متغیرها استفاده از چند عملگر مساوی است:

```
i = j = k = 0;
```

در آخر هر متغیری قبل از استفاده حتما باید تعریف شده باشد. شما می‌توانید تعریف متغیر را در هر قسمتی از کد خودتون قرار بدید.

اپراتورها و عملگرها

برای تعریف یک متغیر کافیست بدینصورت عمل نمایید: اول نام نوع متغیر مورد نظرتون رو تایپ کنید و بعد نامی را برای متغیر انتخاب نموده و در آخر می‌توانید اون رو مقداردهی اولیه نیر بکنید.

استفاده از چند علامت مساوی در مقداردهی اولیه متغیر:

همانند C شما در C# نیز قادر به استفاده از چند علامت مساوی برای مقداردهی به متغیرها هستید. به مثال زیر دقت کنید:

```
i = j = k = 0;
```

خیلی ساده است، این دستور در زمان کامپایل برای CPU به دستوری مانند دستور زیر تبدیل می‌شود:

```
i = 0; j = 0; k = 0;
```

اپراتورهای محاسباتی:

جمع	+
تفریق	-
ضرب	*
تقسیم	/
باقیمانده تقسیم صحیح	%

اپراتورهای منطقی:

عمل ترکیب AND روی بیتها	&
عمل ترکیب OR روی بیتها	|
عمل ترکیب یای انحصاری روی بیتها	^
متمم یک عدد باینری	~
حرکت بیتها به سمت راست	>>n
حرکت بیتها به سمت	<<n

همانند C++، C و جاوا شما در C# اجازه استفاده از عملگرهای کاهش و افزایش را دارید. همچنین شما می‌توانید از دستورات انتساب خلاصه شده نیز استفاده کنید:

```
// Increment and Decrement Operators i = 5;
j = 10;
```

```
x = i++; //x = 5, then i = 6
y = --j; //y = 9 and j = 9
z = ++i; //z = 7 and i = 7
// Combining Arithmetic and Assignment Statements
x = x + 3; //can also be written as:
x += 3; //add 3 to x; store result in x
//also with the other basic operations:
temp *= 1.80; //multiple temp by 1.80
z -= 7; //subtract 7 from z
y /= 1.3; //divide y by 1.3
```

در سی شارپ برای تصمیم گیری از دستور if استفاده می‌شود. شما حتما باید شرط خود را داخل پرانتز قرار دهید. اگر روال اجرایی شرط شما بیش از یک دستور باشد باید از { } استفاده شود. در غیر اینصورت می‌توانید پس از دستور if دستور مورد نظر خود را قرار بدید. اگر می خواهید در صورت عدم برقراری شرط شما دستور یا دستورات خاص دیگری اجرا شود باید از else استفاده نمایید. در این حالت اگر شرط برقرار باشد، مجموعه ای از دستورات و اگر هم شرط برقرار نباشد مجموعه دیگری از دستورات می‌تواند اجرا شود.

```
if ( y > 0 )
   z = x / y;
else
   z = 0;
Console.writeLine("z = " + z);
```

عملگرهای مقایسه ای:

همانند تمامی زبانهای برنامه نویسی سی شارپ نیز دارای عملگرهای مقایسه ای است. اما در نوع نمایش عملگر تفاوت وجود دارد. در سی شارپ از دو مساوی برای مقایسه یکسان بودن استفاده می‌شود. اگر شما به ترکیب شرطها در یک دستور if نیاز داشته باشید می‌تواند از سه عمگر "و"، "یا" و نقیض استفاده کنید. البته این سه عملگر مختص دستور if نیستند.

بزرگتر	>
کوچکتر	<
مساوی	==
مخالف	!=
بزرگتر مساوی	>=
کوچکتر مساوی	<=

عملگرهای منطقی:

عملگر AND منطقی	&&

عملگر OR منطقی	||
عملگر NOT منطقی	~

```
if ( (0 < x) && ( x <= 24) )
   Console.writeLine("Time is up");
```

با استفاده از دستور switch شما می‌توانید مقادیر مختلف امکانپذیر رو برای یک متغیر آزمایش کنید و سپس دستور(های) مناسب رو اجرا نمایید. متغیر شما برای مقایسه باید یک متغیر عددی یا رشته ای باشد که داخل پرانتز قرار گرفته است:

```
switch ( j )
{
 case 12:
  System.out.println("Noon");
  break;
 case 13:
  System.out.println("1 PM");
  break;
 default:
  System.out.println("some other time...");
}
```

با استفاده از دستور break انتهای هر بخش case مشخص میشه. برای تست متغیر با چند مقدار می‌توانید دستورات case رو نوشته و در آخرین دستور، از break استفاده کنید.

توضیحات کد در سی شارپ:

در سی شارپ دو روش برای نوشتن توضیحات وجود دارد. روش تک خطی و روش محدوده ای

```
//C# single-line comment
/* also can go on
for any number of lines*/
```

حلقه ها در سی شارپ

در سی شارپ چهار نوع حلقه وجود دارد که بر اساس نیاز می‌توان از یکی از آنها استفاده نمود. این حلقه ها عبارتند از: `while, do-while, for, foreach`

while: این حلقه بسیار ساده است و تا زمانی که شرط حلقه برقرار باشد، دستورات داخل حلقه اجرا می شوند.

```
i = 0;
while ( i < 100)
   {
   x = x + i++;
   }
```

do-while: این حلقه نیز مانند حلقه `while` می باشد با این تفاوت که حلقه `while` می‌تواند اصلا اجرا نشود اما حلقه `do-while` حداقل یکبار اجرا می‌شود

```
i = 0;
do {
   x += i++;
}
while (i < 100);
```

for:این حلقه یک حلقه ساخت یافته و دارای سه قسمت است: قسمت مقداردهی اولیه، شرط و عملگر تغییر شمارنده حلقه. این سه قسمت توسط ";" از یکدیگر جدا می شوند

```
for (i = 0; //initialize i to 0
     i < 100 ; //continue as long as i < 100
     i++) //increment i after every pass
```

در حلقه بالا شمارنده حلقه از عدد صفر شروع می‌شود. در هر مرحله از اجرا شمارنده `i` با عدد ۱۰۰ مقایسه می‌شود و پس از بررسی اگر `i` کوچکتر از ۱۰۰ باشد بدنه حلقه اجرا می‌شود سپس به `i` مقدار ۱ اضافه می‌شود و ...

تعریف متغیر در حلقه:

در صورت نیاز به تعریف متغیر می‌توانید شمارنده حلقه را در زمان تعریف حلقه تعریف کنید. در این حالت شمارنده حلقه تعریف شده فقط در همان حلقه قابل دسترسی است. به مثال زیر توجه کنید:

```
for (int i =0; i< 5; i++) {
   x[i] = i;
}
System.Console.WriteLine("i=" + i.ToString());
```

استفاده از شمارنده حلقه، خارج از بدنه حلقه ای که متغیر در آن تعریف شده باشد باعث تولید خطا خواهد شد. پس اجرای خط آخر مثال فوق باعث تولید خطا خواهد شد.

استفاده از کاما "," در تعریف حلقه:

با استفاده از کاما می‌توانید بیش از یک شمارنده حلقه در تعریف حلقه داشته باشید. علاوه بر این شما می‌توانید در بخش عملگر نیز بیش از یک عملگر را استفاده نمایید:

```
for (x=0, y= 0, i =0; i < 100; i++, y +=2)
{
   x = i + y;
}
```

کد فوق را می‌توان به شکل زیر نیز نوشت:

```
x = 0;
y = 0;
for ( i = 0; i < 100; i++)
{
   x = i + y;
   y += 2;
}
```

foreach:در این حلقه شما می‌توانید عناصر یک مجموعه را دور بزنید. این مجموعه می تونه شما هر نوع عنصری باشد. برای درک بیشتر به مثال زیر توجه کنید:

```
float[] z = {1.0f, 2.9f, 5.6f};
ArrayList arl = new ArrayList ();
for (int j = 0; j < z.Length ; j++) {
   arl.Add (z[j]);
}
// Sulotion no 1 to access all member of "arl" with for loop
for (j = 0; j < arl.Count ; j++) {
  Console.WriteLine (arl[j]);
}
// Sulotion no 2 to access all member of "arl" with foreach
loop
foreach (float a in arl) {
  Console.WriteLine (a);
}
```

سی شارپ C# زبانی است که بر اساس زبانهای C ++ سی پلاس پلاس، VB ویژوال بیسیک و Java جاوا پیاده سازی شده است. هر دوی زبانهای C# و VB.NET از توابع کتابخانه ای یکسان بهره می برند و

همچنین کد را به یک لایه یکسان کامپایل می‌کنند. هر دو دارای کدهای مدیریت شده (ManagedCode) مثل Garbage Collector هستند. هر دو از کلاسهایی با متدهایی استفاده می‌کنند که اسامی آنها شبیه به موارد مشابه جاوا هست. بنابراین اگر شما با جاوا کار می‌کنید مشکل زیادی در سی شارپ نخواهید داشت.

اشیا در سی شارپ

در سی شارپ همه چیز شی است. شی می‌تواند، داده ها را نگهداری کند، دارای متد باشد که روی آن تاثیر گذار باشد. برای مثال رشته ها (string) الان یک شی است که دارای متدهای مثل این موارد است: ToLowerCase, ToUpperCase, IndexOf ,Insert ,Substring متغیرهای integer, float و Double نیز شی هستند. که هر کدام دارای متدهایی نیز هستند.

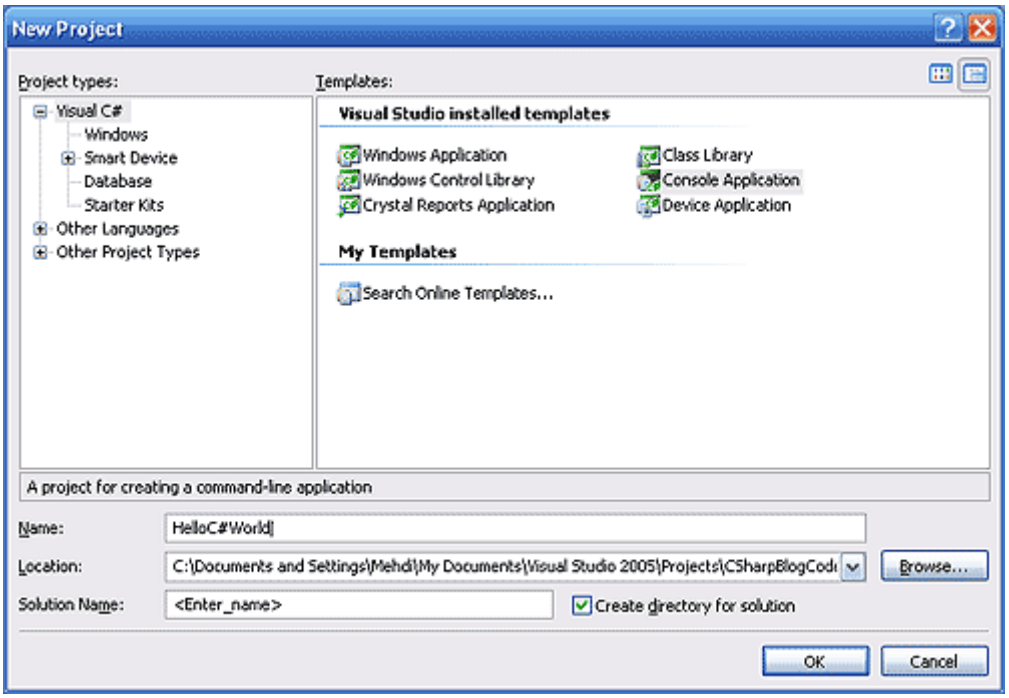
```
string s;
float x;
x = 12.3;
s = x.ToString();
```

زبانهای مدیریت شده و جمع آوری حافظه از دست رفته

سی شارپ و وی بی دات نت هر دو زبانهای مدیریت شده هستند. به این معنی که دارای دو الزام به این شرح هستند: اول هر دو به یک زبان سطح پایین میانی یکسان کدها را کامپایل می‌کنند و CLR برای اجرای کدهای کامپایل شده مورد استفاد قرار می‌گیرد. دوم الزام مهم اینست که زبانهای مدیریت شده حافظه اشغال شده را پس از استفاده و عدم نیاز آزاد می‌کنند. وقتی که قسمتی از حافظه توسط متغیرها، آرایه ها و اشیا اشغال شده باشد اما هیچ مرجعی برای آن وجود نداشته باشد، GB (جمع آوری زباله برگردان واژه به واژه این اصطلاح است) حافظه اختصاص یافته را آزاد می سازد. در این حالت برنامه نویس با خیال راحت و بدون نگرانی از آزاد سازی فضاهای استفاده شده حافظه، می‌تواند از متغیرها و اشیا بدون از بین بردن آنها استفاده کند.

ایجاد یک برنامه سی شارپ

خوب برای اینکه در یک دریا غرق نشویم من پیشنهاد می‌کنم از یک برنامه بسیار ساده شروع کنیم. برای همین میرویم سراغ یک برنامه کنسول. برنامه های کنسول در پنجره Command Prompt اجرا می شوند و هیچ فرم خاصی ندارند. ویژوال استدیو رو اجرا کنید و سپس از منوی File گزینه New Project را انتخاب کنید. در پنجره انتخاب، گزینه Console Application C# رو انتخاب کنید که در تصویر زیر هم می‌توانید مشابه اون رو ببینید:



بعد از انتخاب نوع برنامه، یک ماژول که دارای تعدادی فیلد هست در اختیار شما قرار می‌گیرد. شما می‌توانید کدهای مورد نیاز خود رو بنویسید. برای مثال کد زیر را می‌نویسیم.

```
Console.WriteLine("Hello C# World");
```

خوبه شما اولین برنامه سی شارپ رو با موفقیت نوشتید. می‌توانید اون رو اجرا کنید و نتیجه رو ببینید. البته بعد از اجرا شدن برنامه پنجره Command Prompt رو دیدید که سریع هم بسته شد. نگران نباشید. اگر شما قبلا با C ++ کار کرده باشید می دونید که این پنجره به عنوان پنجره خروجی برنامه عمل می کنه و خوب بعد از اجرای کامل برنامه هم بسته می شه. برای اینکه شما بتونید پنجره خروجی رو باز نگه دارید تا خروجی های برنامه رو ببینید نیاز هست که یک متد برای خواندن اطلاعات از صفحه کلید را صدا بزنید. کد زیر این عمل را انجام می دیه. بنابراین برنامه شما به دو خط تبدیل شد. اولی برای نمایش خروجی و دومی هم برای گرفتن اطلاعات از ورودی.

```
Console.ReadLine();
```

یک برنامه ساده تحت ویندوز توسط سی شارپ

توسط سی شارپ به سادگی می‌توانید رابطهای کاربر ویندوزی را طراحی کنید. این عمل با استفاده از طراح سی شارپ به سادگی صورت می‌گیرد. برای این منظور یک پروژه سی شارپ را آغاز کنید. این مسیر را برای ساخت یک برنامه ویندوزی طی کنید:

`File\New Project\> C# Windows Application`

نام پیش فرض پروژه و فایل آن `WindowsApplication1` است. شما می‌توانید این نام رو تغییر بدید و بعد از اون فرم پیش فرض `Form1.cs` در اختیار شماست که می‌توانید از جعبه ابزار کنترلهای مورد نظر خودتون روی اون قرار بدید. خوب حالا برای نوشتن کد کافیه که روی کنترل مورد نظر دوبار کلیک کنید. در این مثال ما می‌خواهیم که یک کار بسیار ساده انجام بدیم. برای این منظور هم فقط از یک دکمه و یک جعبه متن استفاده می‌کنیم و می‌خواهیم با انتخاب دکمه، متن "به دنیای جادوی سی شارپ خوش آمدید" را در جعبه متن قرار بدیم. پس از دوبار کلیک روی دکمه کد زیر بصورت اتوماتیک ایجاد می‌شود و ما دستورات مورد نیاز برای اعمال تغییرات عنوان شده را می‌نویسیم.

```
private void btnHi_Click(object sender, EventArgs e)
{
    txtMessage.Text = "Hello to the magic world of CSharp";
}
```

برنامه شما آماده اجرا شدن است. پس با استفاده از کلید F5 برنامه رو اجرا کنید و نتیجه عمل رو ببینید.

توضیحاتی در خصوص بقیه دستورات:

دستورات بارگذاری توابع کتابخانه‌ای کلاسهایی که در این برنامه از آنها استفاده شده است: (این کدها به شکل پیش فرض در ابتدای برنامه قرار می‌گیرند و در زمان استفاده از اشیا و کلاسهای زیر گروه آنها نیازی به مشخص نمودن مسیر کامل نیست)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Windows.Forms;
```

کدهایی هم در خصوص ایجاد کنترلها، اضافه شدن به فرم و رویدادهای آن دیده می‌شود:

```
private void InitializeComponent()
{
this.btnHi = new System.Windows.Forms.Button();
this.txtMessage = new System.Windows.Forms.TextBox();
// btnHi
this.btnHi.Location = new System.Drawing.Point(63, 75);
this.btnHi.Name = "btnHi";
this.btnHi.Size = new System.Drawing.Size(145, 23);
this.btnHi.TabIndex = 0;
this.btnHi.Text = "Hello";
this.btnHi.UseVisualStyleBackColor = true;
this.btnHi.Click += new System.EventHandler(this.btnHi_Click);
// txtMessage
this.txtMessage.Location = new System.Drawing.Point(63, 49);
this.txtMessage.Name = "txtMessage";
this.txtMessage.Size = new System.Drawing.Size(145, 20);
this.txtMessage.TabIndex = 1;
.
.
.
}
```

اگر این کدها رو بصورت دستی خودتون تغییر بدید می‌توانید تاثیر اون رو در محیط طراحی برنامه ببینید.

فقط مواظب تغییرات باشید. (اون خطی هم که توپر شده اضافه کردن رویداد به یک شی هست)!

کنترلهای ویندوزی

بسیاری از کنترلهای ویندوزی در زمان استفاده مشابه کنترلهای دکمه و جعبه متن هستند. اگر به تصویر زیر نگاه کنید کنترلهایی که عموما در بسیاری از برنامه ها مورد استفاده قرار می گیرند رو خواهید دید.

همه کنترلهای استفاده شده در تصویر برنامه فوق دارای خصوصیات: Name, Text, Font, ForeColor هستند. شما می‌توانید این خصوصیات را از طریق پنجره خصوصیات یا از طریق برنامه نویسی تغییر دهید.

کلاس فرم که توسط محیط طراحی ویژوال استدیو ساخته شده است دارای یک سازنده به نام Form1 است که در آن متد InitializeComponent صدا زده شده است. این متد خصوصیات کنترلها را تعیین می کند و شما نیز می‌توانید از طریق کد خصوصیات کنترلها را تغییر دهید. البته پیشنهاد می‌شود که مثل من برای تغییر خصوصات کنترلها در ابتدای برنامه از یک متد جدید به نام Init استفاده شود. و این متد پس از متد InitializeComponent صدا زده شود.

برای مثال برای قرار دهی مقادیر اولیه در کنترل کمبو باکس (لیست فروریز) از همین روش استفاده شده و مقادیر در متد init تعریف شده و به لیست کمبو باکس اضافه می شوند.

```
namespace ControlTest
{
public partial class Form1 : Form
{public Form1()
{ InitializeComponent();
init(); }
private void init()
{
cboFontSize.Items.Add("8");
cboFontSize.Items.Add("10");
cboFontSize.Items.Add("12");
cboFontSize.Items.Add("14");
cboFontSize.Items.Add("18");
} } }
```

کنترل برچسب (Label)

برچسب، یک فیلد نمایش متون در ویندوز است. معمولا برنامه نویسها از این کنترل در کنار کنترل جعبه متن (TextBox) استفاده می‌کنند. این کنترل نمی تونه فوکوس فرم را به خودش بگیره. در نتیجه کلیک روی آن یا حرکت Tab تاثیری روی کنترل برچسب ندارد. به هر حال شما می‌توانید بسیاری از خصوصیات این کنترل رو در زمان طراحی یا اجرا تغییر بدید.

خصوصیت	توضیحات
Name	نام کنترل که فقط در زمان طراحی قابل تغییر است
BackColor	رنگ زمینه برچسب
BorderStyle	بدون رسم مرز کنترل، Fixed3D،FixedSingle
Enabled	true یا false اگر false انتخاب شده باشد، کنترل خاکستری می‌شود
Font	تعیین یک فونت جدید
ForeColor	رنگ متن
Image	یک تصویر که در محدوده کنترل نمایش داده می‌شود
ImageAlign	تعیین مکان قرارگیری تصویر
Text	متن برچسب
Visible	true یا false وضعیت رویت کنترل روی فرم

کنترل جعبه متن (TextBox)

جعبه متن یک کنترل با قابلیت تایپ متن است که به شکل تک خطی و چند خطی می‌تواند روی فرم تنظیم شود. شما می‌توانید متن این کنترل رو بگیرید یا متنی رو در کنترل قرار دهید. خصوصیات مهم این کنترل علاوه بر خصوصیات لیست شده برای برچسب عبارتند از:

خصوصیت	توضیحات
Lines	آرایه ای از جنس string که به ازای هر خط یک عنصر دارد
Locked	اگر true انتخاب شده باشد، کاربر امکان تایپ داخل جعبه متن را ندارد
Multiline	falseیاtrue تعیین کننده قابلیت تایپ چند خطی در کنترل
ReadOnly	مشابه خصوصیت Locked کاربر می‌تواند متن رو انتخاب نمایید اما امکان تغییر ندارد
WordWrap	falseیاtrue حالت شکسته شدن خط یا ادامه متن خط

برنامه نمونه: یک جعبه متن و یک کنترل دکمه روی فرم قرار دهید. سپس از کد زیر برای تغییر متن در زمان اجرا استفاده کنید.

```
private void btnSetText_Click(object sender, EventArgs e)
{ txtSample.TextAlign =
System.Windows.Forms.HorizontalAlignment.Right;
txtSample.Text = "سلام به دنیای جادوی سی شارپ";
}
```

خصوصیت TextAlign موقعیت قرارگیری متن رو در جعبه متن مشخص می کنه و همانطور که می بینید متن به سمت راست تراز شده است.

چون ادامه بحثمون یه خورده طولانی هست من ترجیح دادم در قسمتهای کوتاه تر و البته سریعتر اون ها را در سایت قرار بدم. در پستهای بعدی در خصوص دیگر کنترلهای ویندوزی صحبت خواهیم کرد.

راستش حتما در خصوص واژه "لیست فروریز" یه خورده تعجب کرده باشید! من خودم این واژه رو بیشتر از هر واژه فارسی دیگری می‌پسندم. بیشتر نشون می‌ده که این کنترل چی هست و چی کار می‌کنه.

کنترل جعبه انتخاب (Checkbox)

کنترل جعبه انتخاب دارای دو حالت انتخاب شده و انتخاب نشده است. که این دو حالت را می‌توان از طریق پنجره طراحی و از طریق کدنویسی تغییر داد.

این کنترل دارای یک خصوصیت به نام Appearance است که می‌توانید دو مقدار مختلف Appearance.Normal یا Appearance.Button را برای آن انتخاب کنید. اگر حالت Button را انتخاب کرده باشید، کنترل شما به شکل دکمه‌های فشاری (toggle button) دیده خواهد شد و اگر حالت Normal انتخاب شده باشد، کنترل به شکل جعبه انتخابی دیده خواهد شد.

کنترل دکمه (Button)

کنترل دکمه برای گرفتن دستورات کاربر و ارسال به برنامه استفاده می‌شود. وقتی که روی یک دکمه کلیک می‌کنید برنامه رویداد کلیک کنترل را بدست آورده و دستورات نوشته شده را اجرا می کند. برای دسترسی به دستگیره رویداد این کنترل کافیست همانند کنترل جعبه متن روی کنترل دوبار کلیک کنید.

کنترل دکمه رادیویی یا دکمه انتخاب (Radio/Option Button)

دکمه رادیویی یک دکمه دایره ای با قابلیت کلیک است. در یک گروه از دکمه‌های رادیویی فقط یکی می‌تواند انتخاب شود و در صورتی که گروه های بیشتر از یکی در یک فرم موجود باشد باید از کنترل Group Box برای گروه بندی آنها استفاده کرد.

همانند جعبه انتخاب شما می‌توانید برای دکمه‌های رادیویی برای رویداد کلیک آنها برنامه نویسی انجام دهید ولی معمولا این کار انجام نمیشه و برنامه نویسی این کنترلها در رویداد کنترل دیگری مثل دکمه انجام می‌شود.

کنترل جعبه لیست و لیست فروریز (Listbox and Combo Box)

هر دو کنترل جعبه لیست و لیست فروریز حاوی عناصر یک آرایه در لیست هستند. لیست فروریز به برنامه نویس این قابلیت را می دهد که لیست خود را در فضای کوچکی قرار دهد. کاربر می‌تواند در جعبه لیست موارد مختلفی را انتخاب کند در حالی که در لیست فروریز کاربر فقط مجاز به انتخاب یک گزینه است. چند مورد از خصوصیات این دو کنترل در لیست زیر دیده می‌شود:

خصوصیت	توضیحات
Items	مجموعه آیتمهای که در لیست قرار دارند
MultiColumn	اگر true باشد با استفاده از خصوصیت ColumnWidth می‌توان پهنای هر ستون را تعیین کرد
SelectionMode	با تعیین multisimple قادر به انتخاب موارد مختلفی با استفاده از کی بورد خواهید بود و با تعیین multiextended می‌توانید با موس گروه های مختلفی را انتخاب کنید
SelectedIndex	شماره اندیس آیتم انتخاب شده
SelectedIndices	مجموعه آیتمهای انتخاب شده در لیست را برمی‌گرداند
SelectedItem	آیتم انتخاب شده را برمی‌گرداند

مجموعه items

برای اضافه و حذف نمودن آیتمهای یک لیست یا لیست فروریز از مجموعه items استفاده می‌شود. این مجموعه یک لیست آرایه ArrayList است که متدهای اصلی آن را می‌توانید در جدول زیر ببینید:

متد	توضیحات
Add	اضافه نمودن آیتم به لیست
Count	تعداد اعضای لیست
Iten[i]	دسترسی به عضوهای مجموعه
RemoveAt(i)	حذف عضو ام

اگر در لیست چند آیتم انتخاب شده باشد، می‌توان با استفاده از روشهای زیر به مجموعه آیتمهای انتخاب شده دسترسی پیدا کرد:

lsCommands نام کنترل لیست است.

```
ListBox.SelectedIndexCollection it = new
ListBox.SelectedIndexCollection (lsCommands);
ListBox.SelectedObjectCollection so = new
ListBox.SelectedObjectCollection (lsCommands);
```

کنترل منو (Menu)

شما می‌توانید دو نوع منوی مختلف رو به فرمها اضافه کنید:

۱- MenuStrip که در بالای فرمها قرار می‌گیرد.

۲- ContextMenuStrip معمولا از این منو برای کلیکهای راست استفاده می‌شود.

با دوبار کلیک روی گزینه های منو می‌توانید رویداد کلیک اون رو فعال کنید.

کنترل (Tooltip)

کنترل Tooltip یک جعبه رنگی است که در زمان قرارگیری اشاره گر موس روی یک کنترل نمایش داده می‌شود. برای استفاده از این کنترل نیز باید یک نمونه از آنرا روی فرم قرار دهید و سپس کنترلهای مورد نظر رو به اون اضافه کنید. مانند مثال زیر:

```
tips.SetToolTip (btPush, "Press to add text to list box");
tips.SetToolTip (lsCommands, "Click to copy to text box");
```