

بهترین وبلاگ فارسی در ارائه ایبوک

»» IR-E-BOOK.Blogfa.com ««

آموزش ویتروال بیسیک ۶

نام کتاب: Beginning Visual Basic 6

محل نشر: شهرک کاروان (رضویه)

نویسنده: Meisam.Mor

ناشر: IR-E-Book

سال چاپ: 1385

بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

پیشگفتار

کتابی که هم اکنون در حال مطالعه آن هستید برگرفته شده از پندین کتاب دافلی و فارچی می باشد (برای مشاهده ی منابع، به آخرین بخش از این کتاب مراجعه کنید). این کتاب، در مقایسه با دیگر کتابهای موجود در وب، بسیار کاملتر و غنی تر می باشد. این کتاب به صورت کاملاً رایگان ارائه شده است، در نتیجه تعیین کپی رایت برای آن، امری غیر معمول به نظر می رسد. اما در صورت بروز هرگونه مشکل در رابطه با این کتاب و یا در میان گذاشتن نظرات، پیشنهادات و انتقادات، می توانید از طریق زیر با من در ارتباط باشید:



<http://Ir-E-Book.BlogFa.com>



Eshghemany@gmail.com

بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

فصل اول

آشنایی با ویتروال بیسیک

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

آشنایی با تاریخچه‌ی زبان بیسیک

Basic سرنام کلمات Beginner's All-purpose Symbolic Instruction Code

و به معنی زبان همه‌منظوره برای افراد مبتدی است. این زبان برنامه‌نویسی، به دلیل سادگی سافت‌واری، از محبوبیت زیادی برخوردار است. زبان برنامه‌نویسی بیسیک در سال ۱۹۶۴ میلادی، به وسیله‌ی جان کنی و توماس کورتس در دانشکده دارتموث پدید آمد. این زبان، نخستین زبان برنامه‌نویسی نبود ولی هدف آن، فراهم کردن یک زبان ساده برای دانشجویان رشته‌های مختلف بود. تا به امروز نسخه‌های متعددی از زبان بیسیک ارائه شده است که می‌توان به:

QUICK BASIC و GW BASICA, BASICA, ANSI BASIC, QBASIC اشاره کرد.

زبان برنامه‌نویسی بیسیک با ارائه ویژگی‌های تازه‌ای گرفت و دوباره رونق یافت و بیشتر کاربران به دلیل سهولت، این زبان را برای یادگیری انتخاب نمودند.

ویرایش‌های مختلف ویژوال بیسیک

ویرایش آموزشی یا **Learning Edition**، ابتدایی و ساده‌ترین ویرایش ویژوال بیسیک می‌باشد. در این ویرایش، ابزار خاصی در اختیار شما قرار نمی‌گیرد. این ویرایش گزینه‌ای مناسب، برای افراد مبتدی می‌باشد.

ویرایش حرفه‌ای یا **Professional Edition**، برای حرفه‌ای‌ها طراحی شده است. این نسخه، ویرایش آموزشی را با ابزارهای بیشتر به طور کامل تحت پوشش قرار می‌دهد.

ویرایش تجاری یا **Enterprise Edition**، کامل‌ترین و در عین حال محبوب‌ترین ویرایش ویژوال بیسیک می‌باشد. ویرایشی که هم کاربران حرفه‌ای و هم کاربران مبتدی از آن استفاده می‌کنند.



بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

گذری بر ویژگی‌های ویژوال بیسیک

← نزدیکی به زبان ماشین:

ویژوال بیسیک جزو زبانهای سطح بالا^۱ می‌باشد. زبانهای سطح بالا، زبان‌هایی هستند که به زبان محاوره‌ای انگلیسی نزدیک هستند به همین دلیل، باید به وسیله‌ی کامپایلرها و مفسرها^۲ که نوعی مترجم زبان هستند، به زبان ماشین تبدیل شوند.

← رابط برنامه‌نویسی:

رابط‌های برنامه‌نویسی به دو دسته تقسیم می‌شوند: یک: **بر مبنای متن**^۳ دو: **مبتنی بر گرافیک**^۴. زبانهای مبتنی بر متن، کاربر امکان دسترسی مستقیم به گرافیک را ندارد. مانند زبان C که توسط دنیس ریچی در سال ۱۹۷۲ نوشته شد.

زبانهای مبتنی بر گرافیک یا **ویژوال**، امکان دسترسی مستقیم کاربر به گرافیک را فراهم می‌آورند. یعنی اول طراحی و سپس کدنویسی برنامه صورت می‌گیرد. مانند: ویژوال بیسیک

← محیط ساده:

ویژوال بیسیک محیطی بسیار ساده دارد. این محیط که یکی از **محیط‌های توسعه‌یافته‌ی مجتمع** یعنی **IDE**^۵ می‌باشد، به برنامه‌نویسان این امکان را می‌دهد که برنامه‌های تحت ویندوز خود را بدون نیاز به استفاده از برنامه‌های کاربردی دیگر ایجاد، اجرا و خطایابی کنند. این محیط همچنین امکان نوشتن برنامه‌های تحت ویندوز را حتی به کسانی که آشنایی پندانی با برنامه‌نویسی ویندوز ندارند می‌دهد.

=====

مفاهیم بنیادی

قبل از شروع هر کاری لازم است تا با برخی از مفاهیم و اصطلاحات آشنا بشوید. هر چه جلوتر بروید با اصطلاحات بیشتری آشنا خواهید شد.

¹ High-Level Languages

² Interpreters

³ Text-Based

⁴ Graphical Based

⁵ Integrated Development Environment

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

فرم:

در ویژوال بیسیک، یک پنجره، فرم نامیده می‌شود. هر فرم شامل یک نوار عنوان در قسمت بالای خود می‌باشد. این فرم اساس رابط گرافیکی یک برنامه‌ی کاربردی می‌باشد. برقی از برنامه‌ها از یک فرم و برقی دیگر از دو یا بیشتر از دو فرم استفاده می‌کنند.

کنترل:

شیئی که می‌توان در زمان طراحی یا اجرا، برای نمایش یا تغییر داده‌ها روی آن عملیاتی انجام داد. هر یک از این شیء‌ها شامل شمایل گرافیکی^۱ مخصوص به خود هستند.

شیء:

یک ترکیب از کدها و داده‌ها که می‌توان روی آن کار کرد. شیء‌ها شامل مشفصه‌ها و متدها بوده و به وسیله‌ی کلاس تعریف می‌شوند.

مشفصه‌ها:

هر شیء شامل مشفصه‌هایی^۲ است که به طور کامل ظاهر و رفتار آن را تعریف می‌کند. تمامی این مفصصیات و مشفصه‌ها در پنجره‌ی Properties جمع می‌شوند.

آرگومان:

داده‌های ارسال شده به یک روال را آرگومان^۳ گویند. آرگومان می‌تواند یک ثابت، متغیر یا عبارت دیگری باشد.

روال:

بلایکی از کد که می‌تواند از درون یک برنامه‌ی کاربردی فراخوانی شود. یک روال ممکن است برای تعیین محل شیء‌ها روی فرم یا برای مناسبه‌ی اطلاعات از مجموعه‌ی داده‌های کاربر مورد استفاده قرار می‌گیرد.

پروژه:

مجموعه‌ای از فرم‌ها و مدول‌ها که یک برنامه‌ی کاربردی را ایجاد می‌کنند.

Icon¹
Properties²
Argument³

بهترین وبلاگ فارسی در ارائه آیبوک

IR-E-BOOK.Blogfa.com

اهمیت رابط گرافیکی، ظاهر برنامه

اولین و مهمترین مرحله‌ی سافت یک برنامه، **تصمیم‌گیری** می‌باشد؛ تصمیم‌گیری درباره‌ی اینکه از کامپیوتر چه می‌خواهید. اما دومین مرحله که از نظر اهمیت تفاوتی چندانی با مرحله‌ی اول ندارد، **ظاهر برنامه** می‌باشد. ظاهری که می‌تواند به کلی دید کاربر را نسبت به برنامه شما تغییر دهد. برای داشتن ظاهری کاربرپسند، و همچنین محصولی قدرتمند باید سه فاکتور بسیار مهم زیر را در نظر داشته باشید:

۱. سطح مهارت کاربر

۲. عادت‌های کاربر

۳. قابلیت تغییر تنظیمات

۱. اگر شما **سطح دانش و مهارت** کاربر خود را در هنگام طراحی ظاهر برنامه در نظر نگیرید، قطعاً با انتقاد روبه‌رو خواهید شد. معمولاً کاربران مبتدی علاقه‌ی زیادی به سرک کشیدن در برنامه را دارند، در حالی که کاربران حرفه‌ای می‌خواهند هر چه زودتر کارشان با برنامه تمام شود.

۲. **عادت‌های کاربر** را تغییر ندهید یا به عبارتی دیگر استانداردها را رعایت کنید. مثلاً کاربر عادت دارد که Exit را آخرین گزینه از اولین منوی موجود در سمت چپ (منوی File) ببیند یا از کلیدهای ترکیبی Ctrl+C برای کپی کردن استفاده کند. حال اگر شما این عادت‌ها را تغییر بدهید باعث عصبانیت بعضی از کاربران می‌شوید.

۳. رابط کاربر باید **قابل تنظیم** باشد. کاربرها سلیقه‌های مختلفی دارند، در نتیجه باید بتوانند، برنامه‌ی شما را بر حسب سلیقه‌ی خودشان تنظیم کنند. اما پیش فرض باید بهترین تنظیم باشد، تا کاربر مجبور نشود در اولین برخورد با برنامه، به منوی Setting یا Option برود. نحوه‌ی ارتباط برقرار کردن کاربر با برنامه شما بسیار اهمیت دارد، پس سعی کنید ساده‌ترین، شیک‌ترین و جذاب‌ترین محیط کاربری را ایجاد کنید.

اجرای ویژوال بیسیک

ویژوال بیسیک را از طریق زیر اجرا کنید:

Start → All Programs → Microsoft Visual Studio 6.0 → Microsoft Visual Basic 6.0

آشنایی با محیط ویژوال بیسیک

بعد از اجرای برنامه، کادر معاوره‌ای **New Project** به نمایش در می‌آید. این کادر به برنامه‌نویس امکان انتحاب یکی از انواع برنامه‌هایی را می‌دهد که می‌توان در ویژوال بیسیک ایجاد کرد.



این کادر معاوره‌ای دارای سه زبانه^۱ می‌باشد.

- زبانه **New**: برای ایجاد یک پروژه جدید
- زبانه **Existing**: برای باز کردن پروژه‌هایی که از قبل ایجاد شده‌اند.
- زبانه **Recent**: لیستی از آخرین پروژه‌های باز شده یا ایجاد شده را نشان می‌دهد.

تکته:

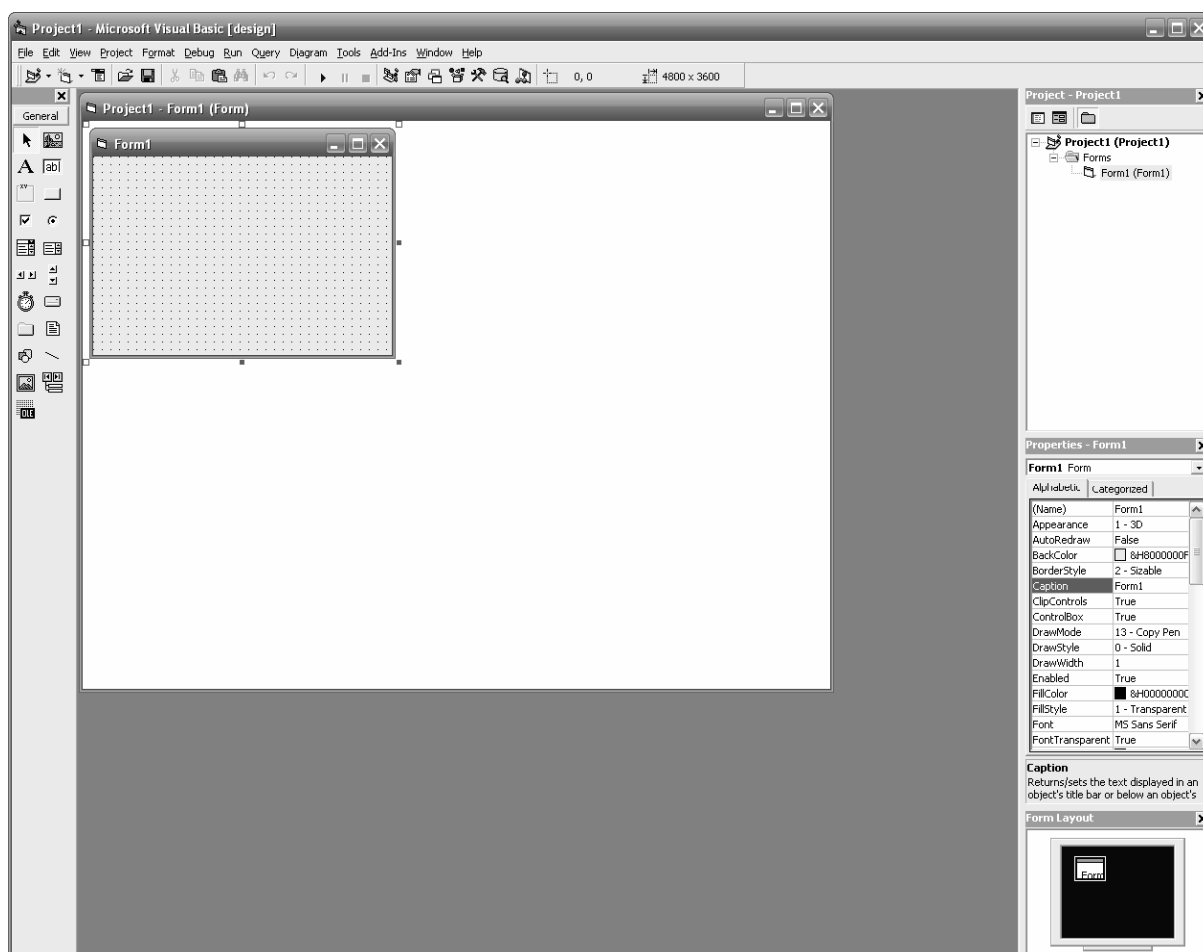
اگر کادر معاوره‌ای **New Project** نمایش داده نشد، احتمالاً کسی گزینه **Don't show this Dialog in the future** را تیک زده است. برای نمایش مجدد این کادر در هنگام شروع برنامه و شروع ایجاد پروژه‌ی جدید، ابتدا به منوی **Tools** رفته، سپس بر روی **Option** کلیک کنید. بعد به زبانه **Environment** رفته و گزینه **Prompt for project** را انتحاب کنید. نوع **Standard EXE** که به طور پیش‌فرض در این کادر انتحاب شده است، به برنامه‌نویس امکان می‌دهد که برنامه‌ی اجرایی استاندارد را ایجاد کند؛ برنامه‌هایی که پسوند **EXE** دارند. برای باز کردن یک پروژه بر روی آیکن مورد نظر، دابل کلیک کرده یا رو آیکن، کلیک کنید، سپس کلید **Enter** یا

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

دکمه‌ی Open را فشار دهید. نوع پیش فرض را باز کنید. با باز شدن پروژه، کادر مشاوره‌ای بسته شده و وارد محیط IDE می‌شویم. پروژه‌ی Standard EXE دارای کادر و پنجره‌های زیر است:

- پنجره‌ی Project
- پنجره‌ی Form Layout
- پنجره‌ی Properties
- پنجره‌ی Form
- جعبه ابزار یا Toolbox
- نوار منو یا Menu Bar
- نوار ابزار یا Toolbar

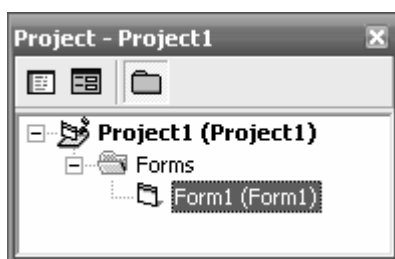


بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

پنجره‌ی پروژه

پنجره پروژه یا **Project Explorer** تمام فایل‌هایی که باعث ایجاد یک برنامه در ویژوال بیسیک می‌شوند را لیست می‌کند. این پنجره یکی از مهم‌ترین ابزارهای مدیریت پروژه می‌باشد. در صورت عدم مشاهده‌ی این پنجره، کلیدهای **Ctrl+R** را فشار دهید یا از منوی **View** گزینه‌ی **Project Explorer** را انتخاب کنید. در قسمت بالای این پنجره، تعدادی دکمه وجود دارد که به عنوان نوار ابزار این پنجره شناخته می‌شوند:



- اولین دکمه از سمت چپ، دکمه **View Code** می‌باشد؛ با کلیک بر روی این دکمه وارد محیط کدنویسی ویژوال بیسیک می‌شوید.
- دومین دکمه **View Object** نام دارد. اگر در قسمت کدنویسی برنامه باشید و این دکمه را کلیک کنید به قسمت طراحی برنامه باز می‌گردید.
- با فعال بودن دکمه‌ی آفر یعنی **Toggle Folders**، تمامی فایل‌های موجود در پنجره‌ی **Project Explorer** به صورت سازماندهی شده در می‌آیند و هر یک از انواع مختلف این فایل‌ها دارای پوشه‌ای اختصاصی می‌شوند.

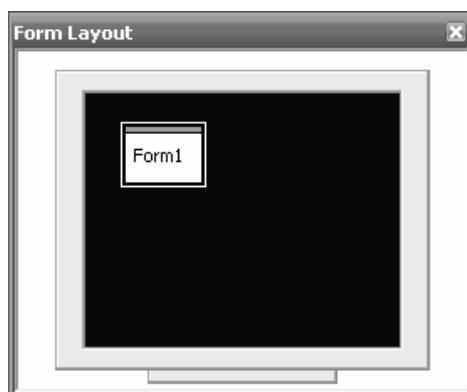
یادآوری:

شما می‌توانید در فضای خالی این پنجره کلیک راست کنید تا با امکانات این پنجره آشنا بشوید. یکی از این گزینه‌ها **Add** می‌باشد که می‌توانید در صورت لزوم، اجزای دیگری را به پروژه اضافه کنید. منظور از اجزا همان فرم‌ها، ماژول‌ها و ... می‌باشند.



پنجره‌ی Form Layout

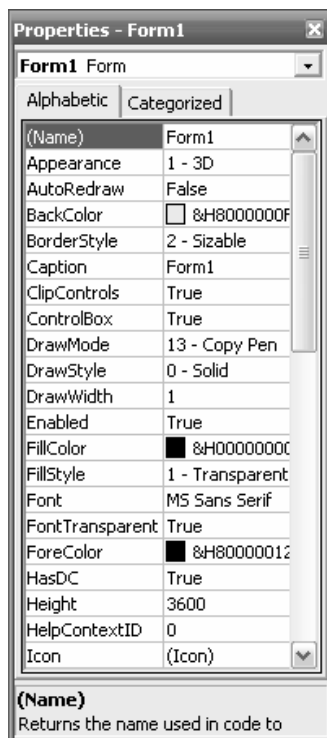
این پنجره محل فرم را به هنگام اجرای برنامه بر روی صفحه‌ی نمایش مشخص می‌کند.



همانطور که می‌بینید این پنجره یک صفحه‌ی نمایش را نشان می‌دهد که در داخل آن محل قرار گرفتن فرم مشخص شده است. شما می‌توانید با کلیک راست بر روی Form1 و انتخاب گزینه Startup Position یکی از انواع مختلف قرارگیری را انتخاب کنید. حالت Manual فرم را در همان قسمتی قرار می‌دهد که شما قبلاً تعیین کرده‌اید. حالت Center Owner فرم را در قسمت مرکز فرم والد قرار می‌دهد. حالت Center Screen فرم را در مرکز صفحه نمایش قرار می‌دهد؛ این گزینه با توجه به رزولوشن صفحه نمایش کار می‌کند. حالت آخر یعنی Windows Default فرم را در هر بار اجرا به صورت تصادفی در یک جایی از صفحه قرار می‌دهد. همچنین شما می‌توانید محل قرارگیری فرم را به صورت دستی تنظیم کنید. برای این کار Form1 را درگ، و در جایی دیگر دراپ کنید. در صورت عدم مشاهده این پنجره از منوی View گزینه‌ی Form Layout Window را انتخاب کنید.

پنجره‌ی مشفصه‌ها

این پنجره ویژگی‌ها و مشفصه‌های فرم یا شیء انتخاب شده را به دو صورت الفبایی و گروهی نشان می‌دهد. در صورت عدم مشاهده‌ی این پنجره کلید F4 را فشار دهید یا از منوی View گزینه Properties Window را انتخاب کنید.



همانطور که در شکل بالا مشاهده می‌کنید در بالای پنجره، جعبه‌ی لیستی وجود دارد که در آن نام شیء یا فرمی که مشفیه‌های آن در این پنجره آورده شده است، نمایش داده می‌شود. داخل این لیست نام تمامی شیء‌ها و همپنین نام فرمی که فعال است، آورده شده است.



پنجره‌ی فرم‌ها

این پنجره، فرم فعال در پنجره‌ی پروژه را با تمام شیء‌های مربوط به آن، در یک **رابط گرافیکی کاربر**^۱ نشان می‌دهد.



جعبه ابزار: این قسمت شامل تمام کنترل‌هایی است که در طراحی برنامه از آنها استفاده می‌کنیم. برقی از این کنترل‌ها مشفیه‌هایی منمصر به فرد و برقی دیگر مشفیه‌های مشترک دارند.

بهترین وبلاگ فارسی در ارائه آیبوک

IR-E-BOOK.Blogfa.com

آیکن	نام کنترل	عملکرد
	Pointer	فعال بودن اشاره گر موس
	PictureBox	کنترلی برای درج تصویر
	Label	از این کنترل برای نمایش متن غیر قابل ویرایش استفاده می شود.
	TextBox	از این کنترل برای متن های قابل ویرایش استفاده می شود.
	Frame	وظیفه ی کنترل فریم دسته بندی ظاهری و منطقی کنترل ها است.
	CommandButton	برای شروع، توقف و پایان یک فرایند از این کنترل استفاده می شود.
	CheckBox	برای ارایه ی انتخاب Yes/No یا True/False به کاربر استفاده می شود.
	OptionButton	دکمه انتخاب برای انتخاب انحصاری یک گزینه از میان چند گزینه به کار می رود.
	ComboBox	ایجاد جعبه ای کشویی یا کرکره ای
	ListBox	برای نمایش لیستی داده ها مورد استفاده قرار می گیرد.
	HScrollBar	از این کنترل برای ایجاد پیمایش گر افقی استفاده می شود.
	VScrollBar	وظیفه ی این کنترل ایجاد پیمایش گر عمودی می باشد.
	Timer	کنترلی که کارهای تکراری را در بازه زمانی تعیین شده انجام می دهد.
	DriveListBox	این کنترل تمامی درایوهای موجود در روی سیستم را نمایش می دهد.
	DirListBox	برای نمایش پوشه های موجود در یک درایو کاربرد دارد.
	FileListBox	برای نمایش فایل های موجود در یک درایو از این کنترل استفاده می شود.
	Shape	از این کنترل برای نمایش اشکال هندسی استفاده می شود.
	Line	کنترلی برای کشیدن خط
	Image	از این کنترل نیز برای نمایش تصویر استفاده می شود.
	Data	کنترلی برای دسترسی آسان به بانک های اطلاعاتی
	OLE	این کنترل امکان دسترسی به داده ها از چندین منبع مختلف را فراهم می کند.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

نوار منو

مکانی است که در اکثر برنامه‌های تحت ویندوز وجود دارد و شامل دستوراتی برای سافت، نگهداری و راه‌اندازی برنامه‌هاست. جدول زیر وظایف هر بخش از منو را به طور خلاصه ارائه کرده است:

وظایف	منوی
باز کردن، ذخیره و چاپ پروژه	File
کپی، انتقال، جستجو، حذف و غیره	Edit
نمونه‌ی نمایش پنجره‌های محیط IDE	View
افزودن فسیله‌هایی مانند فرم‌ها به یک پروژه	Project
تنظیم شیء‌های موجود بر روی فرم	Format
فطایبی	Debug
اجرا، متوقف کردن برنامه و ...	Run
بازیابی داده‌ها از پایگاه داده‌ها	Query
ویرایش و اصلاح در طراحی پایگاه‌های داده	Diagram
ابزارهای VB و بهینه‌سازی محیط کاری	Tools
نصب و حذف برنامه‌های افزودنی	Add-Ins
مرتب کردن و نمایش پنجره‌ها	Window
استفاده از راهنمای برنامه (در صورت نصب MSDN)	Help

نوار ابزار

در پایین نوار منو، ابزاری وجود دارد که دارای دکمه‌های معادل گزینه‌های منوهاست و به وسیله‌ی

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

آنها می‌توان به سرعت برفی از دستورهای موجود در منو را اجرا کرد. برفی از این دکمه‌ها هنگام فعال بودن پنجره‌ی کدنویسی و برفی دیگر در هنگام فعال بودن پنجره‌ی فرم‌ها، فعال می‌باشند.

آیکن	عملکرد
	شروع پروژه‌ای جدید؛ البته بدون استفاده از پنجره‌ی New Project
	برای اضافه کردن فرم، ماثول، کلاس و ... مورد استفاده قرار می‌گیرد.
	نمایش پنجره‌ی Menu Editor؛ برای درج منو در برنامه
	از این دکمه برای بازکردن پروژه‌هایی که از قبلا ایجاد شده‌اند، استفاده می‌شود.
	ذخیره کردن پروژه‌ی در حال کار
	برای انتقال متن یا کنترل از این دکمه استفاده می‌کنند.
	این دکمه در نسخه‌برداری از کنترل‌ها و متن‌ها کاربرد دارد.
	با استفاده از این دکمه می‌توانید محتوای حافظه موقت را به ممل دیگری از برنامه انتقال دهید.
	برای یافتن متن مورد نظر در بخش کدنویسی ویژوال بیسیک، از این دکمه استفاده می‌شود.
	آفرین عمل انجام شده را لغو می‌کند.
	بازگشت به آفرین عمل لغو شده.
	برای اجرای برنامه از این دکمه استفاده می‌شود. (F5)
	متوقف کردن موقت برنامه‌ی در حال اجرا، جهت عیب‌یابی، بازبینی و ...
	این دکمه برنامه‌ی در حال اجرا را، به طور کامل متوقف می‌کند.
	نمایش پنجره‌ی Project Explorer
	نمایش پنجره‌ی مشفصه‌ها
	نمایش پنجره‌ی Form Layout
	نمایش پنجره‌ی Object Browser
	نمایش Toolbox
	ابزاری برای بررسی کردن ساختار و محتویات یک پایگاه داده.
	ابزاری برای ذخیره‌سازی، سازماندهی و اشتراک‌گذاری کامپوننت‌ها.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

اولین پروژه، اولین برنامه

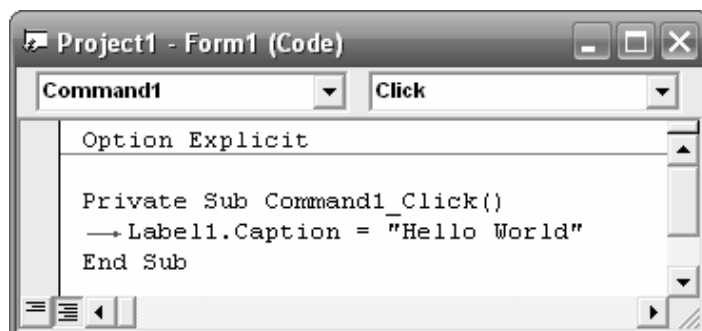
ویژوال بیسیک را اجرا کرده و یک پروژه از نوع استاندارد ایجاد کنید. فرم اصلی یا Form1 را مشاهده می‌کنید. ما در این بخش قصد داریم کارمان را با مثال معروف و کلاسیک Hello World شروع کنیم. از جعبه ابزار کنترل‌های Label و CommandButton را پیدا کنید؛ از هر کدام یکی بر روی فرم قرار دهید؛ برای قرار دادن این شیء‌ها بر روی فرم دو روش وجود دارد:

- انتخاب کنترل مورد نظر ← درگ کردن بر روی فرم
- دابل کلیک بر روی کنترل

حال باید سراغ کدنویسی برنامه برویم. برای رفتن به بخش **کدنویسی** ویژوال بیسیک چند راه وجود دارد:

- دابل کلیک روی شیء
- انتخاب شیء با ماوس و فشار دادن کلید F7
- انتخاب شیء مورد نظر و انتخاب گزینه Code از منوی View
- استفاده از دکمه‌ی موجود در نوار ابزار پنجره‌ی Project Explorer
- کلیک راست بر روی شیء مورد نظر و انتخاب گزینه‌ی View Code
- کلیک راست در پنجره‌ی Project Explorer و انتخاب گزینه‌ی View Code

هم اکنون که زیر را برای **Command1** بنویسید:



بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

برنامه را اجرا کنید و سپس بر روی Command1 کلیک کنید. فواید دید که متن موجود بر روی لیبل تغییر می‌کند. همانطور که گفته شد هر شیء مجموعه‌ای از خواص را با خود یدک می‌کشد. یکی از خواص کنترل Label، عنوان یا Caption آن می‌باشد. همان متنی است که در لیبل نمایش داده می‌شود. اگر برنامه هنوز در حال اجرا می‌باشد، آن را ببندید، تا به محیط طراحی بازگردید. دوباره همان Label را انتخاب کنید و در پنجره‌ی مشخصه‌ها، به دنبال Caption بگردید و مقدار آن را به کلمه‌ای دلخواه تغییر دهید. حال اگر برنامه را اجرا کنید، می‌بینید که در ابتدا کلمه‌ی نوشته شده توسط شما نمایش داده می‌شود و همچنین می‌توانید با فشار دادن دکمه Command1 آن را به Hello World تغییر دهید. اما معنی کدی که نوشتیم چه بود؟ برای پاسخ به این سوال ابتدا باید با مفهوم رویداد آشنا بشوید.

رویداد یک سیگنال تولید شده به وسیله‌ی سیستم عامل در پاسخ به عمل کاربر است. به عنوان مثال، هنگامی که کاربر کلیک می‌کند، رویداد MouseDown به وسیله‌ی سیستم عامل ارسال می‌شود. برنامه‌ی کاربردی فعال، این سیگنال را دریافت کرده و کد مربوط به رویداد MouseDown را اجرا می‌کند. در ویژوال بیسیک قطعه برنامه‌ای به نام Event Handler مسئول این است که رویدادها را کنترل کند. این قطعه برنامه برای هر رویداد کارهای خاصی را انجام می‌دهد و از جمله این کارها، صدا کردن تابع (زیر برنامه) نوشته شده توسط ما می‌باشد. اما مراحل اجرای دستورات نوشته شده توسط ما چگونه می‌باشد؟

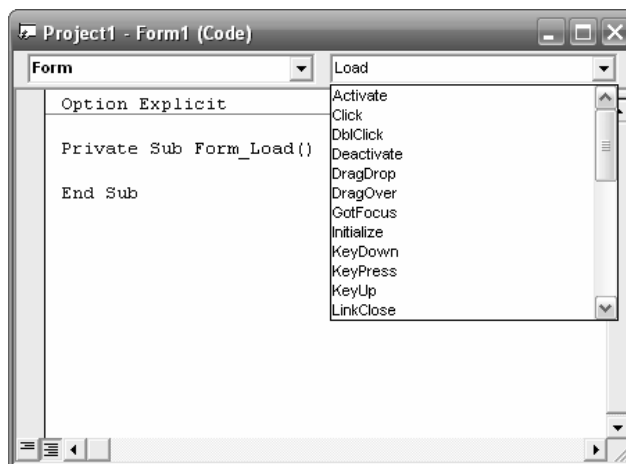
۱. کاربر روی دکمه کلیک می‌کند.

۲. سیستم عامل پیغامی به برنامه ما می‌فرستد و ما را از رخ دادن کلیک بر روی دکمه با خبر می‌کند.

۳. Event Handler پیغام سیستم عامل را دریافت می‌کند و اگر ما زیر برنامه‌ای برای این رفتار نوشته باشیم، آن را صدا می‌کند.

۴. زیر برنامه‌ی ما عنوان Label را تغییر می‌دهد.

شاید پرسید آیا رویدادهای دیگری هم برای یک دکمه اتفاق می‌افتد؟ بله. برای اینکه لیست تمام رویدادهای یک شیء را ببینید کافیهست روی قسمتی که در شکل پایین مشخص شده است کلیک کنید تا بسته به نیاز خود، برای رویدادهای مختلف یک شیء کد بنویسید.



نکته‌ی یک:

روال‌های رویداد از ترکیب نام شیء با نام رویداد به وجود می‌آیند و شکل کلی آن به صورت زیر می‌باشد.

```
Private Sub objectName_event()  
    Statements  
End Sub
```

نکته‌ی دو:

داده‌های رشته‌ای^۱ همیشه بین دو نقل قول ("") قرار می‌گیرند و می‌توانند شامل هر نویسه‌ای باشند. نکته‌ی سه:

مهم‌ترین نکته در مورد عبارت بالا این است که مقدار سمت راست مساوی به عبارت موجود در سمت چپ نسبت داده می‌شود. توجه کنید که مقدار سمت راست عبارت باید با نوع عبارت سمت چپ متناسب باشد، یا اینکه امکان تبدیل آن برای ویروال بیسیک وجود داشته باشد.

نکته‌ی چهار:

اگر درک رویدادها برای شما مشکل می‌باشد می‌توانید آن را برای خود ترجمه کنید. شما می‌توانید از کلمه‌ی کلیدی "اگر" استفاده کنید. مثلاً در مثال بالا می‌توانید بگویید اگر Command1 کلیک^۲ شد، عنوان لیبل یک را به Hello World تغییر بده. (فقط برای مبتدیان)

^۱ String Literal

^۲ منظور همان رویداد کلیک می‌باشد.

ذخیره کردن پروژه

برای ذخیره کردن پروژه کفیسست به منوی فایل رفته و گزینه Save Project را انتخاب نمایید. اگر به محل پروژه‌ی ذخیره شده بروید، خواهید دید که علاوه بر فایل پروژه، چندین فایل دیگر هم وجود دارد. این فایل‌ها عبارتند از:

- **فایل پروژه:**

این فایل با پسوند VBP¹ ذخیره می‌شود و محتوای آن مشخصات پروژه، نوع پروژه، نام فایل‌های فرم و فرم اصلی و ... است.

- **فایل محیط کاری:**

این فایل با پسوند VBW² ذخیره می‌شود و محتوای آن، اطلاعات محیط کاری و فرم‌های پروژه است.

- **فایل فرم:**

این فایل با پسوند FRM ذخیره می‌شود و محتوای آن اطلاعات یک فرم و تمام مشخصات فرم به همراه نام و مشخصات کنترل‌های روی فرم است، در ضمن تمام رویدادها و کدهای مربوط به آن نیز در این فایل ذخیره می‌شوند.

- **فایل تصاویر:**

این فایل با پسوند FRX ذخیره می‌شود و محتوای تصاویری است که روی فرم یا کنترل‌های دیگر از آنها استفاده شده است.

- **فایل‌های دیگر:**

شامل فایل‌های DLL, OCX و ... می‌باشد که در آینده با آنها آشنا خواهید شد.

برنامه‌های جدید (آشنایی با دیگر خصوصیات Label, CommandButton و فرم)

¹ Visual Basic Project
² Visual Basic Workspace

بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

برای درک بهتر این کدها به بخش مشخصه‌های کنترل لیبل بروید.

خاصیت BackColor: تغییر رنگ پس‌زمینه

```
Private Sub Command1_Click()  
    Label1.BackColor = 0  
End Sub
```

خاصیت Font: تعیین نوع و اندازه‌ی قلم

```
Private Sub Command1_Click()  
    Label1.Font.Name = "Tohoma"  
    Label1.Font.Size = "12"  
    Label1.Font.Underline = True  
End Sub
```

Or

```
Private Sub Command1_Click()  
    Label1.Font = "Tohoma"  
    Label1.FontSize = "12"  
    Label1.FontUnderline = True  
End Sub
```

خاصیت ForeColor: تغییر رنگ قلم

```
Private Sub Command1_Click()  
    Label1.ForeColor = &HC00000  
End Sub
```

خاصیت MousePointer: تعیین شکل اشاره‌گر ماوس

```
Private Sub Form_Load()  
    Label1.MousePointer = 5  
End Sub
```

خاصیت MouseIcon: تعیین شکلی دلفواه برای ماوس

اشاره‌گر ماوس یک آیکون ۱۶ x ۱۶ با پسوند ICO است که خودتان هم می‌توانید آن را بسازید. برای نمایش این اشاره‌گر، ابتدا باید خاصیت MousePointer را برابر 99-Custom قرار داده، سپس خاصیت MouseIcon را تغییر بدهید. تا زمانی که اشاره‌گر را دوباره تغییر نده‌اید، این شکل به عنوان اشاره‌گر انجام وظیفه خواهد کرد.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

نکته:

اگر دقت کرده باشید، بعد از نوشتن نام کنترل و زدن نقطه، لیستی از مشفیه‌های آن کنترل برای شما ظاهر می‌شود. برای رسیدن به مشفیه‌ی مورد نظر، کافیهست از کلیدهای جهت‌ی استفاده کنید یا نام اول آن مشفیه را تایپ کنید و سپس **کلید تب** را از روی مشفیه کلید بفشارید.

=====

برای درک بهتر این بخش، مشفیه‌های فرم را بررسی کنید.

خاصیت Caption: تغییر عنوان فرم

```
Private Sub Command1_Click()  
    Form1.Caption = "Changed"  
End Sub
```

خاصیت BorderStyle: تغییر عناصر پنجره

این مشفیه که یکی از مهم‌ترین مشفیه‌های فرم می‌باشد، عناصر پنجره‌ای (نوار عنوان، هاشیه، جعبه‌ی کنترل و غیره) را که فرم خواهد داشت را تعیین می‌کند.

مقدار مشفیه	توضیح
0-None	بدون هاشیه، بدون نوار عنوان و غیر قابل جابه‌جایی ¹
1-Fixed Single	با درگ کردن هاشیه‌ها نمی‌توان تغییر اندازه داد.
2-Sizable	دارای قابلیت تغییر اندازه به وسیله‌ی درگ کردن و وجود دکمه‌های Maximize و Minimize.
3-Fixed Dialog	عدم قابلیت تغییر اندازه و عدم وجود دکمه‌های Maximize و Minimize.
4-Fixed ToolWindow	شبه حالت قبل با این تفاوت که نوار عنوان کوتاه‌تر است و قلم نوار عنوان و دکمه‌ی Close کوچک‌تر است.
5-Sizable ToolWindow	شبه حالت فوق با این تفاوت که با درگ کردن هاشیه تغییر اندازه ممکن است.

¹ البته با برقی از کدها قابل جابه‌جا شدن می‌باشد.

نکته:

تفاوت Fixed Single با Fixed Dialog چیست؟
Fixed Single: در این حالت علاوه بر ظاهر شدن فرم در Taskbar، شما می‌توانید دکمه‌های مکس و مین را به فرم اختصاص دهید.
Fixed Dialog: همانطور که از نامش پیداست، یک کادر معاودهای برای شما ایجاد می‌کند. این کادر در Taskbar دیده نمی‌شود (ویژگی کادرهای معاودهای) و نمی‌تواند خاصیت MaxButton و MinButton را به خود اختصاص دهد.

خاصیت ControlBox: نمایش یا عدم نمایش جعبه‌ی کنترل
اگر برابر False باشد، جعبه‌ی کنترل که شامل دکمه‌های Close، Maximize و Minimize می‌باشد نمایش داده نخواهد شد.

خاصیت MaxButton و MinButton:
اگر برابر False باشند، دکمه‌های Maximize و Minimize نمایش داده نخواهند شد.

خاصیت Movable: عدم جابه‌جایی فرم
اگر برابر False باشد، فرم قابلیت جابه‌جایی ندارد و در محل از قبل تعیین شده‌ی خود ثابت می‌ماند.

خاصیت ShowInTaskbar: نمایش فرم در نوار وظیفه
اگر برابر False باشد، برنامه‌ی کاربردی در نوار وظیفه نمایان نخواهد شد.

خاصیت StartUpPosition: محل نمایش فرم
این مشخصه، دقیقاً کار Form Layout را انجام می‌دهد. اگر قصد تنظیم محل قرارگیری فرم در هنگام اجرای برنامه را دارید، می‌توانید از این گزینه نیز استفاده کنید.

خاصیت WindowState: نمونه‌ی نمایش فرم
اگر قصد دارید که برنامه‌ی شما در هنگام اجرا به صورت تمام صفحه در بیاید، باید مشخصه‌ی WindowState را با Maximized مقداردهی کنید. برای مثال چهار دکمه به فرم اضافه کنید و کدهای زیر را، برای رویداد کلیک آنها بنویسید.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

کلید مورد نظر	کد مربوطه
Close	End
Maximize	Form1.WindowState = 2
Minimize	Form1.WindowState = 1
Restore	Form1.WindowState = 0

نکته‌ی یک:

اگر می‌فواهید قلم تمام اشیائی را که روی فرم قرار می‌دهید یکسان باشند، باید قبل از طراحی ظاهر برنامه، قلم فرم را تغییر دهید؛ چون همانطور که می‌دانید اشیائی که روی فرم قرار می‌گیرند، از برفی از مشفیه‌های فرم تبعیت می‌کنند.

نکته‌ی دو:

در هنگام کدنویسی، اگر چند حرف اول یک کلمه‌ی شناخته‌شده توسط ویژوال بیسیک (مانند نام کنترل‌ها، Property ها، Method ها و غیره) را بنویسیم و **CTRL+Space** را فشار دهیم، اگر VB تنها یک کلمه از حرف اول آنچه را که نوشتیم پیدا کند، بقیه حروف را خود کامل می‌کند و اگر تعدادی کلمه با این مشفیهات پیدا کند، لیست آنها را نمایش می‌دهد که می‌توانید یکی از آنها را انتخاب کنید.

=====

برفی از فصوصیات CommandButton

خاصیت Cancel: اگر برابر True باشد که مربوط به رویداد **Click** دکمه‌ی فرمان، هنگام فشار دادن کلید **ESC** از روی صفحه کلید اجرا می‌شود. فقط یکی از دکمه‌های روی فرم می‌تواند این خاصیت را به صورت فعال داشته باشد.

خاصیت Caption: متن روی دکمه را مشفص می‌کند.

خاصیت Default: اگر برابر True باشد، این دکمه به عنوان دکمه‌ی پیش‌فرض به حساب می‌آید و حتی اگر مکان‌نما هم در روی شیء دیگری باشد و کلید **Enter** فشرده شود، دستورات درون این دکمه اجرا می‌شوند. در اینجا هم فقط یکی از دکمه‌های روی فرم می‌تواند این خاصیت را به صورت فعال داشته باشد.

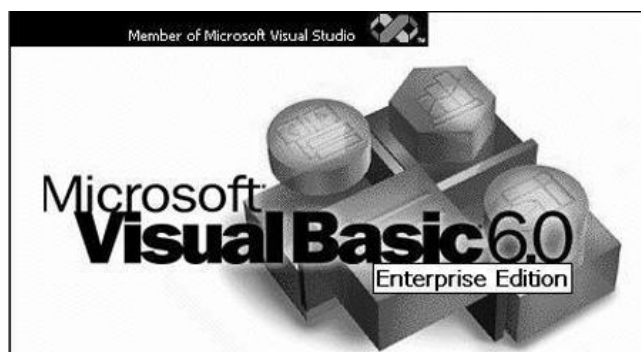
بهترین وبلاگ فارسی در ارائه ایبوک

» **IR-E-BOOK.Blogfa.com** «

خاصیت **Enabled**: اگر برابر False باشد دکمه غیرفعال یا خاموش است و نمی توان رو آن کلیک کرد.

خاصیت **Picture**: فایل bmp. که روی دکمه نمایش داده می شود.

=====



بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

فصل دوم

انواع داده‌ها

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

هر زبان برنامه‌نویسی برای پردازش داده‌ها، به انواع مختلفی از داده‌ها نیاز دارد و ویژوال بیسیک هم از این قاعده مستثنی نیست. ویژوال بیسیک از انواع داده‌های مختلف پشتیبانی می‌کند که می‌توانند نیازهای متعدد برنامه‌نویس را برآورده سازند. به طور کلی، می‌توان داده‌ها را به دو نوع **عددی** و **غیرعددی** تقسیم کرد.

داده‌های عددی

تمام انواع داده‌های عددی در یکی از دو گروه زیر قرار دارند:

اعداد صحیح (Integer): اعداد صحیح بدون نقطه‌ی اعشاری؛ مانند ۱۳۲، ۰، ۶۸۴-.

اعداد اعشاری (Decimal): اعداد با نقطه‌ی اعشاری (ممیز)؛ مانند ۶۵۴/۶، ۰/۰۰۵، ۴۳۲/۶۵-.

اعداد اعشاری، اعداد ممیز شناور هم گفته می‌شود. در تمام اعداد اعشاری باید ممیز اعشار وجود داشته باشد، حتی اگر رقم‌های بعد از آن صفر باشد.

ویژوال بیسیک اعداد اعشاری و صحیح را به روش‌های مختلف ذخیره و با آنها کار می‌کند. با آنکه برای کاربر ۸ و ۸/۰۰ هیچ فرقی ندارد، ولی از نظر ویژوال بیسیک آنها متفاوت هستند. مقدار حافظه‌ای که انواع داده‌های مختلف به خود اختصاص می‌دهند، یکسان نیست. با نگاه کردن به یک عدد هم نمی‌توان گفت که مقدار حافظه اشغال خواهد کرد. با وجود اینکه امروزه، دیگر حافظه یک مشکل کلیدی نیست و شما هم به عنوان برنامه‌نویس نباید زیاده‌نگران آن باشید، ولی همیشه سعی کنید برای داده‌هایتان نوعی را انتخاب کنید که حافظه‌ی کمتری اشغال می‌کند.

در جدول زیر، شش نوع داده‌ی عددی و ویژوال بیسیک، مقدار حافظه‌ی مورد نیاز هر یک و محدوده‌ای که می‌توانند در خود جای دهند، مشاهده می‌کنید. هنگام تعریف داده‌ها این جدول را مدنظر داشته باشید. به عنوان مثال، اگر می‌خواهید با اعداد منفی کار کنید **نباید** از نوع **Byte** استفاده کنید، اما اگر با سن افراد سر و کار دارید، این نوع بهترین انتخاب ممکن است.

نوع داده	فضای ذخیره‌سازی	محدوده‌ی مقادیر
Byte	۱ بایت	0 تا 255
Integer	۲ بایت	-32,768 تا 32,767
Long	۴ بایت	-2,147,483,648 تا 2,147,483,647

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

اعداد منفی: -3.402823E38 تا -1.401298E-45 اعداد مثبت: 1.401298E-45 تا 3.402823E38	۴ بایت	Single
اعداد منفی: -1.79769313486232E308 تا -4.94065645841247E-324 اعداد مثبت: 4.94065645841247E-324 تا 1.79769313486232E308	۸ بایت	Double
922,337,203,685,477.5807 تا -922,337,203,685,477.5807	۸ بایت	Currency

نکته:

برای اعداد اعشاری، بهتر است از نوع Single استفاده کنید.

سایر انواع داده‌ها (داده‌های غیر عددی)

درک داده‌های غیر عددی (برای کسانی که علاقه‌ای به ریاضیات ندارند) آسان‌تر است. یکی از دلایلی که BASIC، علیرغم حضور زبان‌های پیشرفته‌تر همچنان مطرح باقی مانده است، توانایی‌های آن در کار با رشته‌های متنی است. رشته ترکیبی از چند نویسه می‌باشد، که حتی می‌تواند رقم عددی باشد؛ اما نمی‌توان روی آنها **مماسبه** انجام داد. همواره سعی کنید فقط برای اعدادی که نیاز به مماسبه دارند، از انواع عددی استفاده کنید. در جدول زیر انواع داده‌های **غیر عددی** و ویژگی‌های آنها را مشاهده می‌کنید.

نوع داده	مقدار حافظه	محدوده
String (طول ثابت)	طول رشته	از ۱ تا تقریباً ۶۵۴۰۰ نویسه
String (طول متغیر)	طول رشته + ۱۰ بایت	۰ تا ۲ میلیارد نویسه
Date	۸ بایت	از اول ژانویه ۱۰۰ تا ۳۱ دسامبر ۹۹۹۹
Boolean	۲ بایت	True یا False
Object[]	۴ بایت	معدّل شیء تعریف شده
Variant (عددی)	۱۶ بایت	هر عددی تا Double
Variant (متن)	طول رشته + ۲۲ بایت	۰ تا تقریباً ۲ میلیارد نویسه

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

متغیرها

گاهی لازم است در طول برنامه‌تان عددی یا داده‌ای را در جایی ذخیره کنید و روی آن عملیاتی انجام دهید؛ برای این کار به متغیر نیاز دارید. متغیر یا **Variable** مکانی در حافظه است که می‌تواند مقداری را در خود نگه دارد که این مقدار، قابل تغییر است. وقتی مقداری را در یک متغیر قرار می‌دهید، مقدار قبلی آن از بین خواهد رفت.

متغیرها با نامشان شناخته می‌شوند، بنابر این در یک قسمت از برنامه (روال) استفاده از دو متغیر با یک نام مجاز نیست، زیرا ویژوال بیسیک قادر به تشخیص آنها نخواهد بود. قبل از استفاده از یک متغیر باید آن را اعلان (Declare) کنید. اعلان یک متغیر، یعنی نام‌گذاری آن و تعیین نوع مقداری که می‌تواند بگیرد.

نکته:

متغیرها فقط می‌توانند از همان نوعی که اعلان شده‌اند مقدار بگیرند؛ البته به‌استثنای متغیرهای Variant که می‌توانند از تمام انواع داده‌ها مقدار بگیرند.

قوانین ایجاد متغیرها

به‌دلیل اینکه نام‌گذاری متغیرها به عهده‌ی برنامه‌نویس است، باید قواعد نام‌گذاری آنها را بدانید:

- نام متغیر باید با یکی از حروف الفبا شروع شود.
- استفاده از حروف و اعداد در نام متغیرها مجاز است.
- نام یک متغیر می‌تواند تا ۲۵۵ نویسه طول داشته باشد.
- سعی کنید تا حد امکان از حروف خاص (غیرالفبایی-عددی) استفاده نکنید. بدین ترتیب دیگر نیازی نیست نگران باشید که کدام حروف خاص مجازند و کدام حروف غیرمجاز.
- فاصله در نام متغیرها مجاز نیست.
- علاوه بر قواعد الزامی فوق، سعی کنید هنگام نام‌گذاری متغیرها نکات زیر را هم رعایت کنید.
- در نام متغیرها از پیشوندهایی استفاده کنید که نوع آن را مشخص کند. بدین ترتیب دیگر نیازی نیست که کدام برای اطلاع از نوع یک متغیر به قسمت اعلان متغیرها مراجعه کنید.
- جدول زیر پیشوند انواع داده‌های متعارف ویژوال بیسیک را نشان می‌دهد.

جدول پیشوند نام متغیرها

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

پیشوند	نوع داده	مثال
bln	Boolean	blnButtonEnabled
byt	Byte	bytLenght
cur	Currency	curSales98
dte	Date	dteOverdue
dbl	Double	DbScientificAmt
int	Integer	intYear1998
lng	Long	lngWeatherDistance
obj	Object	objWorksheetAcct99
sng	Single	sngSales1stQte
str	String	strFirstName
vnt	Variant	vntValue

نکته‌ی یک:

در جدول بالا از نام‌های بامعنی استفاده شده است، این کار سبب قابل فهم‌تر شدن برنامه خواهد شد، در نتیجه به مستندسازی کمتری نیاز خواهد داشت.

نکته‌ی دو:

برای جدا کردن قسمت‌های نام متغیر از حروف بزرگ استفاده کنید. مانند: intTotal

اعلان متغیرها

برای اعلان یک متغیر از کلمه‌ی کلیدی **Dim** استفاده می‌شود. قبل از استفاده از یک متغیر حتما باید آن را اعلان کرد. البته VB اجازه می‌دهد که این قاعده‌ی کلی را زیر پا بگذارید، ولی تفل از این قاعده می‌تواند به سردرگمی منجر شود. الزام یا عدم الزام به اعلان متغیرها را می‌توانید در منوی Tools\Option، زبانه‌ی Editor و گزینه‌ی Require Variable Declaration مشخص کنید. اگر این گزینه انتخاب شود، در قسمت تعاریف پنجره‌ی کد، کلمه‌ی Option Explicit به صورت پیش فرض نوشته می‌شود.

تقریباً می‌توان گفت که این عبارت مفهف All Variables Used Must Be Explicitly Declared می‌باشد. این دستور به ویژوال بیسیک می‌گوید که تمامی متغیرهای مورد استفاده، باید

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

صریحا تعریف شوند. در چنین حالتی، هرگاه نام یک متغیر را اشتباه بنویسید، ویژوال بیسیک آن را به شما گوشزد خواهد کرد؛ اما اگر این گزینه غیرفعال باشد، ویژوال بیسیک اشتباه در نوشتن نام یک متغیر را **متغیر جدیدی** تلقی کرده و به کار خود ادامه خواهد داد. در این حالت تمام متغیرهایی که اعلان نشوند، از نوع Variant در نظر گرفته خواهند شد.

شکل کلی استفاده از دستور Dim برای اعلان یک متغیر به صورت زیر می باشد:

Dim VarName [As DataType]

Dim نام متغیر [As نوع داده]

یا

Dim VarName نوع پسوند:

Dim A%

نکته ی یک:

در مثال بالا (Dim A%)، از **پسوند عددی** استفاده کردیم. وقتی در یک برنامه، عددی را صریحا می نویسد، ویژوال بیسیک مناسب ترین نوع را برای آن برمیگزیند، ولی گاهی لازم است داده ی عددی مورد استفاده از نوعی باشد که شما دارید، نه آنچه که ویژوال بیسیک تعیین می کند. در چنین مواردی می توانید نوع داده را صریحا به ویژوال بیسیک معرفی کنید. این کار با استفاده از پسوند نوع داده یا **Data-type Suffix** امکان پذیر است. جدول زیر انواع پسوندهای عددی در ویژوال بیسیک را نشان می دهد.

پسوند	نوع داده
%	Integer
&	Long
!	Single
#	Double
@	Currency

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

نکته ی دو:

اگر به مثال بالا دقت کرده باشید، As DataType درون کروشه قرار گرفته است. یعنی این قسمت Optional یا اختیاری می باشد. هرگاه As DataType ذکر نشود، VB به طور خودکار آن را از نوع Variant تعریف می کند. اگر از نوع داده خود بی اطلاع هستید از نوع Variant استفاده کنید. پس عملاً دو دستور زیر معادلند:

Dim x as Variant

Dim x

نکته ی سه:

هنگام استفاده از مقداری که شامل تاریخ و زمان هستند از علامت # در ابتدا و انتهای این مقادیر، استفاده کنید. ویروال بیسیک از تمام قالب های تاریخ و زمان پشتیبانی می کند. به مثال های زیر توجه کنید:

#July 4, 1776#

#9:02 pm#

#16:09:40#

#1-2-2005#

#5-Dec-99#

نکته ی چهار:

نوع داده ی Boolean برای مواردی مناسب است که فقط دو مقدار مخالف هم دارید. مشخصه ی Enabled کنترل ها، از این نمونه است. سعی کنید هنگام نام گذاری متغیرهای Boolean از سوالاتی استفاده کنید که بتوان به آنها جواب بلی یا خیر داد.

نکته ی پنج:

نوع داده ی Variant می تواند هر مقداری (بجز رشته های با طول ثابت) را در خود جای دهد. زمانی از این نوع داده استفاده کنید که از قبل، دقیقاً نمی دانید با چه نوع داده ای سروکار خواهید داشت.

نکته ی شش:

اگر امکان انتخاب بین دو یا چند نوع انتخاب را دارید، نوع داده ای را انتخاب کنید که، کمترین میزان اشغالی حافظه را داشته باشد. این کار باعث بارگذاری سریع تر برنامه در حافظه و بهینه شدن آن می شود.

نکته ی هفت:

رشته‌ای که طول آن صفر باشد، رشته‌ی Null نامیده می‌شود. کلمه‌ی رزرو شده‌ی VBNullString در VB معادل رشته‌ی تهی می‌باشد. این کلمه می‌تواند به جای دو علامت نقل قول استفاده شود. نکته‌ی هشت:

می‌توان تعاریف متغیرها را با پراسازی توسط کاما در یک دستور Dim ترکیب کرد. ولی اگر متغیرها از انواع گوناگون باشند باید As DataType را نوشت:

```
Dim a1, Total, Sum As Integer
```

```
Dim a1 As Integer, strFirstName As String, t As Boolean
```

متغیرهای رشته‌ای

نوع داده‌ی String برای دو نوع رشته با طول ثابت و متغیر به کار می‌رود. در ویژوال بیسیک، تعریف رشته با طول متغیر، به صورت زیر می‌باشد.

```
Dim strCityName As String
```

هر دوی این متغیرها می‌توانند رشته‌هایی با طول متفاوت را در خود نگه دارند. مثلاً، اگر ابتدا در متغیر strCityName رشته‌ی "Tehran" و سپس رشته‌ی "Barare" را ذخیره کنیم، این متغیر طول خود را متناسب با آن تغییر خواهد داد. ولی اگر بخواهید رشته‌هایی با طول ثابت ایجاد کنید، باید طول رشته را مشخص کنید:

```
Dim VarName As String * Length
```

مانند:

```
Dim strZipCode As String * 5
```

یعنی متغیر strZipCode هیچگاه بیش از ۵ نویسه را نمی‌تواند در خود ذخیره کند. اگر شما سعی کنید رشته‌ای که بیش از ۵ نویسه دارد را در strZipCode ذخیره کنید، VB تنها ۵ نویسه‌ی اول آن را در strZipCode قرار می‌دهد و مابقی را نادیده می‌گیرد.

مقدار دادن به متغیرها

بعد از اعلان یک متغیر، می‌توان داده‌ها را در آن ذخیره کرد. ساده‌ترین راه مقدار دادن به یک متغیر، استفاده از دستور **انتساب**^۱ می‌باشد. شکل کلی این دستور چنین است:

¹ Assignment Statement

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

VarName = Expression

که در آن VarName نام متغیر یا خاصیت یک کنترل است و Expression یکی از مواد زیر می باشد:

- عبارات مناسبی یا ریاضی

$x = 2 * n + i$ یا $Sum = 12 + 34$

- یک مقدار (ثابت، متغیر و ...)

$x = a$ یا $Even = 2$ یا $Name = "Meisam"$

- مشفیه ی یک کنترل

$LblTitle.Caption = "Esteghlal Ghahreman"$ یا $x = LblTitle.Enabled$

نکته ی یک:

مشفیه ی کنترل ها از نوع Variant می باشند ولی ویژوال بیسیک هنگام ذخیره کردن آنها در یک متغیر، نوع آنها را تبدیل خواهد کرد.

نکته ی دو:

همانطور که گفته شد مهم ترین نکته در باره ی یک عبارت آن است که مقدار سمت راست عبارت به متغیر سمت چپ آن نسبت داده می شود. توجه کنید که مقدار سمت راست عبارت باید با نوع متغیر سمت چپ عبارت متناسب باشد.

نکته ی سه:

در ویژوال بیسیک برای حفظ سازگاری با بیسیک های قدیمی، می توان از کلمه ی کلیدی **Let** برای مقدار دادن به متغیرها استفاده کرد؛ پس در عمل دو دستور زیر معادل یکدیگرند:

Let intSales = 4582

intSales = 4582

نکته ی چهار:

هر کنترل ویژوال بیسیک دارای یک مشفیه ی پیش فرض است که اگر مشفیه را ذکر نکنید، ویژوال بیسیک آن را در نظر خواهد گرفت. مشفیه ی پیش فرض برپسب، Caption می باشد، یعنی عبارت زیر با مثالی که در چندین خط بالاتر زدیم، برابر است:

LblTitle = "Esteghlal Ghahreman"

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

با اینکه این روش ساده تر است، ولی از وضوح برنامه می‌کاهد. بنابراین سعی کنید نام مشخصه را ذکر کنید.

دید و مدت عمر متغیر

مقدار هر متغیر در یک محدوده‌ی فاص از برنامه قابل دیدن است که به آن محدوده، محدوده‌ی دید متغیر می‌گویند. تعیین محدوده، بیش از هر چیز به محل تعریف متغیر بستگی دارد. اگر بخواهیم از یک متغیر در همه‌ی فرم استفاده کنیم، باید آن را در قسمت Declarations تعریف کنیم؛ یعنی محلی که Option Explicit وجود دارد. دو کلمه‌ی کلیدی Public و Private نیز در تعیین محدوده‌ی دید موثرند.

وقتی متغیری را با کلمه‌ی کلیدی Public تعریف می‌کنیم، می‌توانیم از آن در تمام برنامه استفاده کنیم، ولی متغیری که با کلمه‌ی کلیدی Private تعریف می‌کنیم، فقط در همان فرم یا ماژول، قابل استفاده می‌باشد.

مدت زمانی را که یک متغیر، مقدار فعلی خود را حفظ می‌کند، مدت عمر متغیر می‌گویند. متغیرهایی که در بخش Declarations تعریف می‌کنیم طول عمری برابر با فرم دارند؛ یعنی با ایجاد فرم، ایجاد و با از بین رفتن فرم، از بین می‌روند (حافظه را به سیستم برمی‌گردانند). متغیرهایی که داخل یک تابع تعریف می‌شوند، طول عمری برابر با طول عمر تابع دارند؛ یعنی با از بین رفتن تابع از بین می‌روند. به مثال زیر توجه کنید؛ (در این مرحله باید بدانید از چه کنترل‌هایی باید استفاده کنید).

```
Private Sub cmdRun_Click()  
Dim X As Integer  
X = X+1  
Label1.Caption = X  
End Sub
```

برنامه را اجرا کنید و یک بار روی cmdRun کلیک کنید.

متغیر X تعریف می‌شود و به طور پیش فرض به آن مقدار صفر داده می‌شود. سپس مقدار آن برابر با یکی بیشتر از مقدار قبلی می‌شود ($X = X+1$) و در لیبل نمایش داده می‌شود. نتیجه همان است که می‌خواستیم. عدد ۱ نشان داده می‌شود. اما اگر یک بار دیگر کلیک کنیم عدد چند می‌شود. تغییری نمی‌کند! چرا؟ به این دلیل که، پس از رسیدن به خط End Sub متغیر X به پایان عمر خود می‌رسد و حافظه‌ای که در اختیار خود گرفته بود به سیستم عامل بازمی‌گرداند.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

وقتی دوباره کلیک می‌کنیم، متغیر دوباره تعریف می‌شود و مقدار اولیه صفر می‌گردد و ...
حال از کلمه‌ی کلیدی **Static** به جای Dim استفاده کنید و برنامه را اجرا کنید:

```
Private Sub cmdRun_Click()
```

```
Static X As Integer
```

```
X = X+1
```

```
Label1.Caption = X
```

```
End Sub
```

احتمالا تعجب فواید کرد. عددی که تغییری نمی‌کرد، حالا با هر بار کلیک، مقدارش یک واحد بیشتر از مقدار قبلی می‌شود. نتیجه‌گیری: دید متغیرهای Static مانند Dim و طول عمر آنها برابر فرم است. با این ترتیب با خروج از تابع، متغیر از بین نمی‌رود و با بازگشت بعدی به تابع، مقدار قبلی خود را حفظ و در نظر می‌گیرد.

نمایش متن روی فرم و کادر تصویر

برای نمایش متن روی یک فرم یا کادر تصویر، از متد **Print** که بعد از نام فرم یا کادر تصویر قرار می‌گیرد، استفاده کنید. برای ارسال خروجی به چاپگر، از این متد روی شیء **Printer** استفاده کنید. شکل کلی متد **Print** به صورت زیر است:

```
[Object.] Print [outputlist] [{; | ,}]
```

آرگومان **Object** اختیاری است و در صورتی که نوشته نشود، خروجی این متد روی فرم جاری نمایش داده می‌شود. آرگومان **outputlist** نیز متنی است که روی فرم یا کادر تصویر ظاهر می‌شود. به عنوان مثال، عبارت‌های زیر پیامی را چاپ می‌کنند:

- چاپ روی فرم

```
Form1.Print "The Method is Print Not PrintForm "
```

- چاپ درون کادر تصویر

```
Picture1.Print "Have a go"
```

- چاپ روی فرم

```
Print "Cool"
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

• شیء Printer

Printer.Print "Kakol Pesar"

نکته ی یک:

اگر پس از دستور Print، هیچ عبارتی (مقدار ثابت یا متغیر) نوشته نشود، سبب فواید شد که روی شیء جاری، یک **خط خالی** نشان داده شود.

نکته ی دو:

پس از متد Print می توان یک یا چند عبارت نوشت. برای جدا کردن این عبارات ها می توان از علامت کاما (,) یا سمی کولن (;) استفاده کرد. در صورتی که چند عبارت را با کاما جدا کنید، شیء جاری به ناهیه های ۱۴ تایی تقسیم می شود و هر عبارت در یک ناهیه نمایش داده می شود. اگر عبارتی، به بیش از ۱۴ فضا برای نمایش نیاز داشته باشد، فضایی مورد نیاز را از ناهیه ی بعدی در اختیار فواید گرفت.

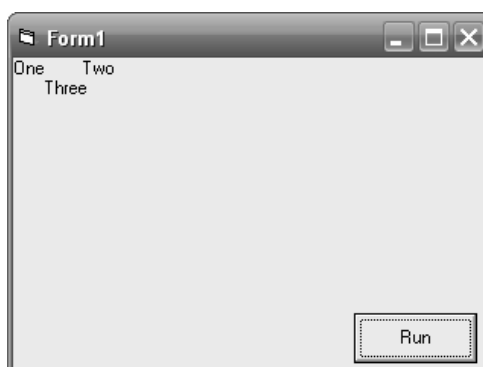
نکته ی سه:

معمولا هر متد Print، سبب انتقال خودکار مکان نما به سطر بعدی می شود، ولی اگر در انتهای یک متد Print از علامت سمی کولن (;) استفاده کنید، فروبی متد Print بعدی در **همان سطر** ظاهر فواید شد.

تابع TAB

این تابع نیز در متد Print، برای **تغییر موقعیت نمایش** مورد استفاده قرار می گیرد و به صورت کلی TAB(n) است. به عبارت دیگر، می توان با استفاده از تابع TAB محل شروع نمایش داده ای را بر روی صفحه ی نمایش تعیین کرد. مثال زیر را در نظر بگیرید:

```
Print "One"; Tab(10); "Two"; Tab(5); "Three"
```



بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

نکته ی یک:

اگر مقدار n کوچکتر از شماره ی ستون محل جاری مکان نما باشد، داده ی مورد نظر از ستون n به بعد، در سطر بعد نمایش داده خواهد شد.

نکته ی دو:

اگر پس از تابع Tab() علامت کاما قرار گیرد، تابع Tab نادیده گرفته شده و داده در سطر بعد نمایش داده خواهد شد.

تبدیل انواع داده ها

ابتدا مثال زیر را امتحان کنید: نتیجه ۵ می باشد.

```
Private Sub Command1_Click()  
Print 2 + "3"  
End Sub
```

حال این کد را تست کنید: نتیجه ۲۳ می شود.

```
Private Sub Command1_Click()  
Print "2" + "3"  
End Sub
```

در حالت دوم ویتوال پیسیک به جای جمع کردن دو عدد، آنها را به هم الحاق کرده است. برای تبدیل انواع مختلف داده ها به یکدیگر، از توابع زیر استفاده کنید:

CBool (Expression)	تبدیل به نوع Boolean
CByte (Expression)	تبدیل به نوع Byte
CDate (Expression)	تبدیل به نوع Date
CDbl (Expression)	تبدیل به نوع Double
CDec (Expression)	تبدیل به نوع Decimal
CInt (Expression)	تبدیل به نوع Integer
CCur (Expression)	تبدیل به نوع Currency

CSng (Expression)	تبدیل به نوع Single
CStr (Expression)	تبدیل به نوع String
CVar (Expression)	تبدیل به نوع Variant
CLng (Expression)	تبدیل به نوع Long

مانند:

```
Print CInt("2") + "3"
```

=====

ثابت‌ها (Constants)

متغیرها تنها روش ذخیره‌ی اطلاعات در حافظه نیستند. روش دیگر، استفاده از **ثابت‌هاست**. ثابت خانه‌ای از حافظه می‌باشد که مقدار آن در طول برنامه ثابت می‌ماند و نمی‌توان مقدار آن را تغییر داد. ثابت‌ها اغلب برای جایگزینی مقادیری که به خاطر سپردن آنها مشکل است یا برای پرهیز از نوشتن مکرر رشته‌های طولانی استفاده می‌شوند. ثوابت با کلمه‌ی کلیدی **Const** تعریف می‌شوند که شکل کلی آن به صورت زیر می‌باشد:

```
[Private/Public] Const ConstantName [As ConstantType] = Value
```

همانطور که می‌دانید بخش اول خط بالا یعنی **Private/Public** مربوط به نحوه‌ی دیره شدن ثابت تعریف شده، توسط دیگر اجزای پروژه می‌باشد. **ConstantName** نام ثابت، **ConstantType** نوع داده‌ی ثابت و **Value** مقداری است که در ثابت ذخیره می‌شود. مانند:

```
Const Pi As Single = 3.14159
Const Name As String = "Meisam"
Const BirthDate As Date = #05/15/1932#
```

```
Option Explicit
Const Pi As Single = 3.14159
Private Sub Command1_Click()
    Print Pi
End Sub
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

نکته ی یک:

اگر قبل از کلمه ی کلیدی Const چیزی نوشته نشده باشد، مقدار پیش فرض Private می باشد.

نکته ی دو:

اگر ConstantType ذکر نشود نوع ثابت Variant خواهد بود.

نکته ی سه:

یک ثابت عمومی (Public Const) فقط می تواند در یک ماژول تعریف شود.

متد cls

به کمک این متد می توان متن یا گرافیک موجود روی فرم یا کادر تصویر را در زمان اجرا پاک کرد. این متد تاثیری روی کنترل ها و تصاویر مربوط به مشغله ی Picture فرم نخواهد داشت. شکل کلی این متد به صورت زیر است:

Object.cls

```
Private Sub cmdClear_Click()  
    Picture1.Cls  
    Form1.Cls  
End Sub
```

نکته:

آرگومان Object **اختیاری** بوده و شبیه متد Print عمل می کند. امتحان کنید!

عملگرها

عملگرها کاراکترها یا نمادهای خاصی هستند که برای انجام عملیات خاص روی متغیرها مقادیر ثابت، عبارات و ... مورد استفاده قرار می گیرند. عملگرها به سه دسته تقسیم می شوند:

۱. محاسباتی و رشته ای

۲. مقایسه ای یا رابطه ای

۳. منطقی

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

عملگرهای مناسباتی و رشته‌ای

عملگر	مفهوم	مثال	نتیجه
^	توان	2^3	8
*	ضرب	$2*3$	6
/	تقسیم	$6/2$	3
+	جمع	$2+3$	5
-	تفریق	$6-3$	3
Mod	باقیمانده	11 Mod 3	2
\	تقسیم صحیح	$11 \setminus 3$	3
& یا +	الاق رشته‌ها	"Moni" & "tor"	Monitor

نکته‌ی یک:

عملگر Mod فقط برای اعداد صحیح است و اگر از اعداد اعشاری استفاده کنید، ویژوال بیسیک ابتدا آنها را به عدد صحیح تبدیل کرده و سپس باقیمانده را محاسبه خواهد کرد.

نکته‌ی دو:

عملگر تقسیم صحیح، خارج قسمت صحیح تقسیم را برمی‌گرداند و از باقیمانده‌ی تقسیم صرف‌نظر می‌کند.

نکته‌ی سه:

عملگر + دو کار متفاوت انجام می‌دهد: **جمع معمولی و الاق (ترکیب) رشته‌ها**. این عملگر با توجه به محلی که مورد استفاده قرار گرفته است (بین دو عدد یا دو رشته) واکنش مناسب را نشان می‌دهد.

نکته‌ی چهار:

هنگام ترکیب رشته‌ها، ویژوال بیسیک هیچ‌چیز به آنها اضافه نخواهد کرد. بنابراین اگر می‌خواهید بین دو رشته یک فاصله وجود داشته باشد باید خودتان آن را اضافه کنید. (استفاده از نقل قول)

=====

تقدم عملگرها

ویژوال بیسیک اعمال ریاضی را به ترتیب زیر انجام می دهد:

۱. پرانتز
۲. تفریق یکنانی
۳. ضرب و تقسیم اعشاری
۴. تقسیم صحیح
۵. Mod
۶. جمع و تفریق

اگر از پرانتزها استفاده نکنید، ویژوال بیسیک همیشه ابتدا توان، سپس ضرب و تقسیم و بعد از آن جمع و تفریق را انجام خواهد داد. مانند:

$$20/2*3$$

در این عبارت به دلیل اینکه ضرب و تقسیم دارای تقدم یکسان هستند، ویژوال بیسیک ابتدا تقسیم را انجام داده و سپس حاصل تقسیم را در ۳ ضرب خواهد کرد. نکته ی یک:

اگر می خواهید ترتیب انجام محاسبات را تغییر دهید باید از پرانتزها استفاده کنید. در پرانتزهای تو در تو، ویژوال بیسیک از داخلی ترین زوج پرانتز شروع کرده و رو به بیرون حرکت می کند. ویژوال بیسیک قبل از هر کاری (3-8) را محاسبه خواهد کرد. مانند:

$$(10 + 2 - (8 - 3)) + 1$$

عملگرهای مقایسه ای یا رابطه ای

در جدول زیر عملگرهای رابطه ای ویژوال بیسیک را مشاهده می کنید. این عملگرها هیچگونه **عملیات ریاضی انجام نمی دهند**، بلکه داده ها را مقایسه می کنند. با این عملگرها برنامه هوشمندتر خواهد شد و خواهد توانست داده ها را مقایسه کرده و بر اساس نتایج آن، عملکرد مناسب را در پیش گیرد. عملگرهای رابطه ای روی عبارت ها، متغیرها، مشفصه ها، کنترل ها یا ترکیبی از آنها عمل می کنند. با توجه به تنوع داده ها در ویژوال بیسیک، این برنامه نویس است که باید تصمیم بگیرد داده ها را چگونه

بهترین وبلاگ فارسی در ارائه آیبوک

IR-E-BOOK.Blogfa.com

مقایسه کرده و چگونه نتیجه‌گیری کند. هنگام استفاده از عملگرهای رابطه‌ای، ممکن است حالت خاصی پیش آید و آن Null بودن یکی از اجزای مقایسه است. در این حالت، ویژوال بیسیک مقدار Null برمی‌گرداند نه True یا False.

عملگر	توضیح	مثال	نتیجه
>	بزرگ‌تر از	$6 > 3$	درست
<	کوچک‌تر از	$5 < 11$	درست
>=	بزرگ‌تر یا مساوی	$23 >= 23$	درست
<=	کوچک‌تر یا مساوی	$4 <= 21$	درست
=	تساوی	$7 = 2$	نادرست
<>	نامساوی	$3 <> 3$	نادرست

عملگرهای رابطه‌ای، علاوه بر مقادیر عددی، می‌توانند رشته‌ها را هم مقایسه کنند. در هنگام مقایسه‌ی رشته‌ها به نکات زیر توجه داشته باشید:

- **حروف بزرگ، کوچکتر از حروف کوچک هستند**، یعنی "IRAN" قبل از "iran" قرار خواهد گرفت.
- **حروف الفبا به همان ترتیبی که هستند مقایسه می‌شوند**، یعنی "A" کوچک‌تر از "B" است و نام "Ahmad" قبل از "Ali" قرار خواهد گرفت.
- **رقم‌ها کوچکتر از حروف هستند**، یعنی "3" از "Three" کوچکتر خواهد بود.

ویژوال بیسیک هنگام مقایسه‌ی رشته‌ها به بزرگ یا کوچک بودن حروف توجه دارد یعنی بین آنها تفاوت قایل خواهد شد. برای درک بهتر این نکات، فقط کافی است بدانید که ویژوال بیسیک رشته‌ها را مانند یک لغت‌نامه مرتب خواهد کرد. به دستورهای مقایسه‌ای زیر توجه کنید:

"abcdef" > "ABCDEF"
"Yes!" < "Yes?"
"PC" <> "pc"

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

"Computers are fun!" = "Computers are fun!"

"Books, Books, Books" >= "Books, Books"

نکته:

هنگام مقایسه‌ی دو مقدار، باید نوع داده‌های شما با هم سازگار باشند. مثلاً نمی‌توان یک عدد را با رشته مقایسه کرد.

عملگرهای منطقی

عملگرهای منطقی یا Logical Operators عبارت‌های موردنظر را بیت به بیت با یکدیگر مقایسه کرده و دو ارزش منطقی T یا F (درست یا نادرست) را برای آنها مشخص می‌کنند.

P Imp Q	P Eqv Q	P Xor Q	P Or Q	P And Q	Not P	Q	P
T	T	F	T	T	F	T	T
F	F	T	T	F	F	F	T
T	F	T	T	F	T	T	F
F	T	F	F	F	T	F	F

مثال:

Dim a, b, c As Boolean

a = Not b a = True

a = b And c a = False

a = b Or c a = True

a = b Xor c a = True

a = b Imp c a = True

a = b Eqv c a = False

عملگر **Not**، روی یک عبارت عمل می‌کند و آن را نقیض می‌کند. اگر عبارت دارای ارزش T باشد، نقیض آن دارای ارزش F است و برعکس.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

عملگر **And**، بر روی دو عبارت عمل می‌کند. ارزش نتیجه وقتی T است که ارزش دو عبارت T باشد و در غیر این صورت، F است.

عملگر **Or** روی دو عبارت عمل می‌کند و ارزش نتیجه وقتی F است که ارزش هر دو عبارت F باشد و در غیر این صورت، ارزش آن T است.

عملگر **Xor** روی دو عبارت عمل می‌کند و ارزش نتیجه وقتی T است که ارزش یکی از دو عبارت، T و ارزش دیگری F باشد.

عملگر **Eqv** روی دو عبارت عمل می‌کند و ارزش نتیجه وقتی T است که هر دو عبارت دارای ارزش یکسان باشند یعنی هر دو دارای ارزش T یا F باشند.

عملگر **Imp** نیز روی دو عبارت عمل می‌کند و ارزش نتیجه در صورتی F است که ارزش عبارت دوم F باشد و در غیر این صورت، T است.

ترکیب عملگرهای رابطه‌ای و منطقی

از لحاظ تئوری، شش عملگر رابطه‌ای و بی‌شمار بیسیک قدرت کافی برای هر نوع مقایسه‌ای را دارند، ولی می‌توان قدرت آنها را با استفاده از عملگرهای منطقی افزایش داد. برای مثال به جدول زیر توجه کنید.

عملگر	مثال	نتیجه
And	(2<3) And (4<5)	True
Or	(2<3) Or (6<7)	True
Xor	(2>3) Xor (7>4)	False
Not	Not (3=3)	False

کاربرد And و Or بیشتر از سایر عملگرهاست. با Xor می‌توان دو گزینه‌ی انحصاری را مقایسه کرد. اگر هر دو گزینه Xor درست باشد، کل عبارت نادرست خواهد شد. عملگر Not هم برای نقیض یک عبارت است. عبارت زیر را در نظر بگیرید:

$curSales * sngCommission > curHighSales / 10$

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

ویژوال بیسیک کدام عمل را زودتر انجام خواهد داد؟ آیا ابتدا sngCommission را با curHighSales مقایسه کرده و حاصل آن را در curSales ضرب و سپس بر ۱۰ تقسیم خواهد کرد؟ البته این کار بی معنی است، چون حاصل مقایسه True یا False است و نمی توان روی آن اعمال ریاضی انجام داد. در جدول زیر تقدم عملگرها را مشاهده می کنید. با توجه به این جدول درس نتیجه ی عبارت فوق ساده خواهد بود.

ترتیب	عملگر
۱	^
۲	- (تفریق یکانی)
۳	/ , *
۴	\
۵	Mod
۶	- , +
۷	= , <> , < , > , <= , >=
۸	Not, And, Or, Xor, Eqv, Imp

کنترل Text Box (جعبه متن)

از این کنترل برای وارد کردن یک مقدار خاص به وسیله ی کاربر استفاده می شود و مهم ترین خاصیت های آن به شرح زیر می باشند:

خاصیت Locked: غیر قابل ویرایش بودن

اگر برابر True باشد، کاربر مجاز به تغییر متن داخل جعبه در زمان اجرا نمی باشد. در این حالت شما فقط می توانید از مفتویات جعبه کپی بگیرید.

خاصیت MaxLength: ایجاد محدودیت

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

اگر مقدار آن صفر باشد هیچ محدودیتی در طول متن وجود ندارد؛ ولی اگر مثلا، مقدار آن ۵ انتخاب شده باشد کاربر نمی‌تواند بیش از ۵ کاراکتر را وارد کند.

خاصیت MultiLine: حالت چندخطی

اگر برابر True باشد، در زمان اجرا با زدن کلید Enter در جعبه‌متن، مکان نما به خط بعد منتقل می‌شود. به عبارت دیگر، به محض رسیدن متن به هاشیه‌ی TextBox، متن به خط بعدی شکسته می‌شود. ولی اگر False باشد تمام متن در یک خط نوشته می‌شود.

خاصیت PasswordChar: به رمز درآوردن مفتویات جعبه‌متن

این مشخصه، نویسه‌ای که در جعبه‌متن نمایش داده می‌شود را تعیین می‌کند. به عنوان مثال، اگر بخواهید در کادر گذرواژه، ستاره نمایش داده شود، در پنجره‌ی مشخصه‌ها، خاصیت PasswordChar را با * مقداردهی کنید. بدین ترتیب، به ازای تایپ هر نویسه‌ای در جعبه‌متن، یک ستاره نمایش داده خواهد شد.

نکته:

اگر جعبه‌متن در حالت چندخطی یا MultiLine قرار گیرد، این خاصیت عمل نخواهد کرد.

خاصیت ScrollBars: ایجاد پیمایش‌گر

اگر متن ورودی بیشتر از طول یا عرض جعبه‌متن باشد، به ScrollBar نیاز خواهیم داشت. اگر مقدار آن 0 - None باشد، پیمایش‌گری نخواهیم داشت، اگر برابر 1 - Horizontal باشد، پیمایش‌گری افقی و اگر برابر 2 - Vertical باشد، پیمایش‌گری عمودی خواهیم داشت. حالت 3 - Both هم ترکیب حالت اول و دوم می‌باشد.

نکته:

این خاصیت فقط با True بودن MultiLine عمل خواهد کرد.

خاصیت Text: متن جعبه‌متن

متن وارد شده در کنترل جعبه‌متن، در مشخصه‌ی Text قرار دارد. به طور پیش‌فرض، می‌توان در یک جعبه‌متن تا 2048 نویسه را وارد کرد. در صورتی که مشخصه‌ی MultiLine با True مقداردهی کنید، می‌توانید متن وارد شده را در چند سطر نمایش داده و حداکثر تا 32k نویسه وارد کنید.

=====

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مثال‌ها:

۱. در برنامه‌ی زیر، ارتفاع و پهنای فرم با اعدادی که ما تعیین می‌کنیم، مشخص می‌شود. برای این کار دو لیبل، دو جعبه‌متن و یک دکمه مانند شکل زیر بر روی فرم قرار دهید.



```
Private Sub Command1_Click()  
    Form1.Height = Text1.Text  
    Form1.Width = Text2.Text  
End Sub
```

Or

```
Private Sub Command1_Click()  
    Dim txtHeight, txtWidth As Integer  
    txtHeight = Text1.Text  
    txtWidth = Text2.Text  
    Form1.Height = txtHeight  
    Form1.Width = txtWidth  
End Sub
```

حال می‌فواهیم کمی برنامه را تغییر دهیم. بررسی کنید که تفاوت بین دو کد زیر چه می‌باشد (در تکه کد دوم پیزی در جعبه‌متن ننویسید).

```
Private Sub Command1_Click()  
    Form1.Height = Text1.Text  
    Form1.Width = Text2.Text  
End Sub
```

With

```
Private Sub Command1_Click()  
    Text1.Text = Form1.Heigh  
    Text1.Text = Form1.Width  
End Sub
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

۲. برنامه‌ای بنویسید که با تایپ متن در TextBox، همزمان عنوان یا متن موجود در کنترل Label را نیز تغییر دهد.

```
Private Sub Text1_Change()  
    Label1.Caption = Text1.Text  
End Sub
```

Change رویدادی است که هنگام تغییر دادن یک شیء اجرا می‌شود. بنابراین Caption لیبل ما با Text موجود در Text1 مقداردهی می‌شود.

۳. برنامه‌ای بنویسید که با اجرای آن سال تولد کاربر را به صورت شمسی در ورودی دریافت کند، سپس سال تولد او را به سال میلادی تبدیل نماید و پس از محاسبه سن او را بر حسب سال در سال ۱۳۸۵، سال میلادی را همراه با سن او چاپ نماید (یک دکمه و بجهت متن بر روی فرم قرار دهید).

```
Private Sub Command1_Click()  
    Dim SY, ADY, AGE As Integer  
    SY = Text1.Text  
    ADY = SY + 621  
    AGE = 1385 - SY  
    Print ADY, AGE  
End Sub
```

برای تبدیل سال شمسی به سال میلادی، سال شمسی را با ۶۲۱ جمع می‌کنیم. در مثال بالا SY محل نگهداری سال تولد شمسی و ADY محل نگهداری سال تولد میلادی در نظر گرفته شده است.

۴. برنامه‌ای بنویسید که با اجرای آن تعداد روزهای ماه از ورودی دریافت گردد، سپس مقدار آن بر حسب ساعت، دقیقه و ثانیه محاسبه و چاپ شود (از شما انتظار داریم با نگاه کردن به کد متوجه بشوید به چه کنترل‌هایی نیاز داریم).

```
Private Sub Command1_Click()  
    Dim Day, Hours, Minute, Second As Single  
    Day = Val(Text1.Text)  
    Hours = Day * 24  
    Minute = Hours * 60
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

Second = Minute * 60
Print Hours, Minute, Second
End Sub

نکته:

در حالت عادی مجعبه متن با نوع داده‌ی Variant کار می‌کند، یعنی هم با اعداد و هم با رشته‌ها کار می‌کند، اما اگر با داده‌های عددی سروکار دارید بهتر است، برای بالا بردن دقت برنامه، قبل از محل ورودی برنامه، از کلمه‌ی کلیدی Val استفاده کنید. Val تابعی است که یک رشته‌ی عددی را به عدد معادل آن تبدیل می‌کند.

=====

بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

فصل سوم

کنترل برنامه

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

در این فصل با سافتهای تصمیم، تابع MsgBox و InputBox و همچنین کنترل‌های Check Box، Option Button و Frame و ... آشنا خواهید شد.

سافتهای تصمیم

یکی از مهم‌ترین سافتهایی که در همه‌ی زبان‌های برنامه‌نویسی مورد استفاده قرار می‌گیرد، سافتهای شرطی یا سافتهای تصمیم‌گیری می‌باشد. برای مثال در یک برنامه‌ی کوچک، وقتی بخواهند دو عدد را مقایسه کنند و بزرگترین را درست بیاورند، از همین سافتهای استفاده می‌کنند. به طور کلی، به مجموعه‌ای از دستورالعمل‌ها که امکان انتقاب و تصمیم‌گیری از بین یک یا چند موضوع را به ما می‌دهند "سافتهای تصمیم" گفته می‌شوند.

دستور If

یکی از مهم‌ترین دستورهای زبان برنامه‌نویسی ویژوال بیسیک، دستور If است، که به چند روش مختلف می‌توانیم از آن استفاده کنیم. شکل کلی دستور If چنین است:

If (شرط) Then

مجموعه دستورات ویژوال بیسیک

End If

در این نوع اگر شرط درست باشد، آنگاه مجموعه دستوراتی که در آن بلاک نوشته شده است، اجرا می‌شوند، وگرنه هیچ اتفاقی نمی‌افتد.
نکته‌ی یک:

برای وضوح برنامه بهتر است که بدنه‌ی If را جلوتر از End If If بنویسید تا تشخیص دستورهای درون If از دستورهای بیرون آن ساده‌تر باشد.
نکته‌ی دو:

پراتتهای اطراف شرط If الزامی نیستند ولی به خوانایی این دستور کمک می‌کنند.

شکل دوم

شکل دوم از سافتهای تصمیم، نیازی به End If ندارد و اگر شرط درست باشد، مجموعه دستورات بعد از Then اجرا می‌شوند وگرنه هیچ اتفاقی نمی‌افتد:

IF (شرط) Then مجموعه دستورات ویژوال بیسیک

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مردم در زندگی روزمره خود به دفعات از If استفاده می کنند:

If (کارم زودتر تمام خواهد شد) Then (زودتر سرکار بروم)
If (نمره ای قبولی فوادم گرفت) Then (درسم را خوب بفوانم)

شکل سوم

در شکل سوم، اگر شرط درست باشد آنگاه مجموعه دستورات ۱ اجرا می شوند، در غیر این صورت مجموعه دستورات ۲ اجرا می شوند:

If (شرط) Then
مجموعه دستورات ویژوال بیسیک ۱
Else
مجموعه دستورات ویژوال بیسیک ۲
End If

شکل چهارم

نوع چهارم از سافتهای تصمیم هم نیازی به End If ندارد و اگر شرط درست باشد مجموعه دستورات ۱ اجرا می شوند، در غیر این صورت مجموعه دستورات ۲ اجرا می شوند.

If شرط Then دستور ۱ Else دستور ۲

سافتار ElseIf

اگر بفوایم با استفاده از If شرطهای گوناگونی را چک کنیم، می توانیم از ElseIf استفاده کنیم. دستورهایی If ... ElseIf ... End If (حتی در ساده ترین شکل) پیچیده اند و امکان بروز خطا در آنها بسیار زیاد است. همچنین این سافتار، باعث کم شدن خوانایی برنامه، سردرگمی برنامه نویس و طولانی شدن برنامه می گردد. به همین منظور پیشنهاد می شود که از دستور Select Case استفاده کنید که در آینده با آن آشنا فواید شد.

مثالها:

تعیین بزرگ ترین عدد

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

```
Private Sub Command1_Click()  
Dim A, B As Integer  
A = Val(Text1.Text)  
B = Val(Text2.Text)  
If (A > B) Then  
    Print A  
Else  
    Print B  
End If  
End Sub
```

اگر امتحان کرده باشید وقتی دو عدد مساوی به برنامه می‌دهید، برنامه نتیجه‌ی درستی به شما ارائه نمی‌دهد. برای رفع این مشکل کافیسست برنامه‌تان را به صورت زیر تغییر دهید:

```
1. Private Sub Command1_Click()  
2. Dim A, B As Integer  
3. A = Val(Text1.Text)  
4. B = Val(Text2.Text)  
5. If (A = B) Then  
6.     Print "Same Numbers"  
7.     Exit Sub  
8. End If  
9.     If (A > B) Then  
10.         Print A  
11.     Else  
12.         Print B  
13.     End If  
14. End Sub
```

خروج زودرس

شاید این سوال برای شما پیش آمده باشد که منظور از Exit Sub در خط هفتم چیست؟ بسته به شرایط یک روال، ممکن است بخواهید روال را از حالت معمول، زودتر پایان یابد. برای این منظور باید از دستور Exit استفاده کنید. شکل کلی این دستور به صورت زیر می‌باشد:

Exit Sub | Function | Do | For

بهترین وبلاگ فارسی در ارائه آیهوک

IR-E-BOOK.Blogfa.com

فقط عمودی بین کلمات، نشان‌دهنده‌ی آن است که در هر دستور Exit یکی از این کلمات را می‌توان به‌کار برد. به عنوان مثال، برای خروج از یک تابع باید از دستور Exit Function و برای خروج از یک روال رویداد باید از دستور Exit Sub استفاده کنید. در مثال بالا، ما هم از این دستور استفاده کردیم؛ زیرا بعد از دریافت دو عدد مساوی، دیگر نیازی به چک کردن خط‌های بعدی برنامه (خط‌های ۹ تا ۱۳) نداریم.

تعیین روز سال

فرض کنید در روز D از ماه M هستیم؛ برنامه‌ای بنویسید که با اجرای آن M و D در ورودی دریافت گردند، سپس مشخص شود که در روز چندم سال هستیم. مثلاً اگر $D = 28$ باشد و $M = 10$ باشد، آنگاه در روز سیصد و چهارم سال هستیم.

```
Private Sub Command1_Click()  
Dim D, M, N As Integer  
D = Val(Text1.Text)  
M = Val(Text2.Text)  
If (M <= 6) Then  
    N = (M - 1) * 31 + D  
Else  
    N = (M - 1) * 30 + 6 + D  
End If  
Text3.Text = N  
End Sub
```

چون تعداد روزهای هر یک از ماه‌های ۴ ماه اول سال ۳۱ روز و ۵ ماه بعد ۳۰ روز است لذا M با ۴ مقایسه گردیده که مشخص شود، آیا در نیمه‌ی اول سال هستیم یا در نیمه‌ی دوم سال؛ و چون به جز ماه جاری تعداد روزهای بقیه‌ی ماه‌ها کامل است، لذا یک واحد از M کسر گردیده و به صورت M-1 منظور گردیده است که نشانگر تعداد ماه‌های قبل است.

تمرین:

برنامه‌ی بالا رو طوری تغییر دهید که هر دو جعبه‌متن (Text1, Text2) فقط قابلیت دریافت دو عدد را داشته باشند (استفاده از یکی از خصوصیات جعبه‌متن)؛ همچنین اگر عدد موجود در جعبه‌متن اول

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

بالتر از ۳۱ و عدد موجود در جعبه‌متن دوم بالاتر از ۱۲ بود، برنامه از روال خارج شود یا پیغامی مناسب نمایش دهد.

زوج یا فرد بودن عدد

1. Private Sub Command1_Click()
2. Dim k As Integer
3. k = Val(Text1.Text)
4. If (k Mod 2 = 0) Then
5. Label1.Caption = "The number is Even"
6. Else
7. Label1.Caption = "The number is Odd"
8. End If
9. Text1.Text = ""
10. Text1.SetFocus
11. End Sub

در فط نهم برنامه اعلام کردیم که بعد از پایان یافتن، مراحل تشخیص عدد زوج و فرد، مقویات جعبه‌متن را خالی کن و بعد از آن Focus را به جعبه‌متن انتقال بده؛ یعنی شیء را در حالت انتخاب قرار بده. برای درک بهتر این موضوع (Focus)، چندین کنترل روی فرم قرار دهید و بعد برنامه را اجرا کنید. خواهید دید که با هر بار فشار دادن کلید Tab از روی صفحه کلید، Focus به کنترل بعدی انتقال می‌یابد. برای تغییر دادن ترتیب Focus گرفتن، باید مقدار TabIndex هر کنترل را تغییر دهید. اگر فواستید کنترلی با زدن کلید تب از سوی کاربر، Focus دریافت نکند، صفت TabStop آن کنترل را برابر False قرار دهید.

تعیین مثبت یا منفی بودن عدد

```
Private Sub Command1_Click()  
Dim a As Integer  
a = Val(Text1.Text)  
If (a > 0) Then  
    Label1.Caption = "Positive"  
Elseif (a < 0) Then  
    Label1.Caption = "Negative"  
Else
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

```
Label1.Caption = "Zero"  
End If  
End Sub
```

بررسی گذرواژه

```
Private Sub Command1_Click()  
Dim Pass As String  
Pass = txtPassword.Text  
If Pass = "Meisam" Then  
    Beep  
    Label1.Caption = "Welcome"  
Else  
    Label1.Caption = "Try again"  
    Pass = ""  
End If  
End Sub
```

نکته:

Beep دستوری است که یک صدای بیپ در اسپیکر کامپیوترتان ایجاد می‌کند.

=====

تابع MsgBox()

به کمک این تابع می‌توان پیغامی را به کاربر نمایش داد و پاسخ وی را دریافت کرد. این پیغام کادری است که شامل یک آیکن، یک پیام و حداقل یک دکمه است؛ شکل کلی تابع MsgBox() چنین است:

Variable = MsgBox (StrPrompt [, intStyle] [, strTitle])

([عنوان پنجره "], [عبارت تعیین کننده نوع و تعداد کلید], "پیام") = MsgBox = متغیر

تابع MsgBox() یک آرگومان اجباری (StrPrompt) و دو آرگومان اختیاری (intStyle و strTitle) دارد. از پیشوند نام این آرگومان‌ها می‌توانید حدس بزنید که نوع آنها چیست. مقدار برگشتی تابع، در متغیری صحیح قرار می‌گیرد. وقتی کاربر پیام ظاهر می‌شود، اجرای برنامه موقتاً متوقف می‌شود تا کاربر دکمه‌ای را کلیک کند. بعد از آن، اجرای برنامه از دستور بعد از تابع

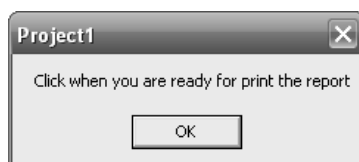
بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

MsgBox() از سرگرفته خواهد شد. فرض کنید در برنامه‌ای قبل از چاپ گزارش، کاربر باید آمادگی خود را اعلام کند. دستور ساده‌ی زیر این کار را انجام می‌دهد:

MsgBox ("Click when you are ready for print the report")

در شکل زیر، این کادر پیام ساده‌ای را نمایش می‌دهد. در این حالت دیگر مقدار برگشتی تابع اهمیت چندانی ندارد و می‌توان از روش دستوری این تابع استفاده کرد که نیازی به مقدار برگشتی ندارد.



نکته:

اگر آرگومان اختیاری دوم (strTitle) قید نشود، ویژوال بیسیک از نام پروژه در عنوان پیام استفاده می‌کند.

مقادیر برگشتی:

برای اینکه بدانیم عکس‌العمل کاربر نسبت به کادر پیام چیست، یا متوجه بشویم که کاربر روی کدام دکمه کلیک کرده است باید از مقادیر برگشتی استفاده کنیم. در جدول زیر مقادیر برگشتی کادر پیام را مشاهده می‌کنید:

مقادیر	نام ثابت	مفهوم
1	vbOK	کاربر روی Ok کلیک کرده است.
2	vbCancel	کاربر روی Cancel کلیک کرده است.
3	vbAbort	کاربر روی Abort کلیک کرده است.
4	vbRetry	کاربر روی Retry کلیک کرده است.
5	vbIgnore	کاربر روی Ignore کلیک کرده است.
6	vbYes	کاربر روی Yes کلیک کرده است.
7	vbNo	کاربر روی No کلیک کرده است.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مقدار برگشتی MsgBox() زمانی مفهوم پیدا می‌کند که دکمه‌های کادر پیام بیش از یکی باشد. برای کنترل بیشتر دکمه‌های کادر پیام باید از آرگومان افتیاری اول (intStyle)، استفاده کنید؛ که به سه بخش تقسیم می‌شوند:

بخش اول: تعداد و نوع کلید

مقدار	نام ثابت	مفهوم
0	vbOKOnly	نمایش کلید OK
1	vbOKCancel	نمایش کلیدهای OK و Cancel
2	vbAbortRetryIgnore	نمایش کلیدهای Abort و Retry و Ignore
3	vbYesNoCancel	نمایش کلیدهای Yes و No و Cancel
4	vbYesNo	نمایش کلیدهای Yes و No
5	vbRetryCancel	نمایش کلیدهای Retry و Cancel

مثال:

Response = MsgBox ("Ready to print?", 1)

Or

Response = MsgBox ("Ready to print?", vbOKCancel)

بخش دوم: تعیین آیکن کادر پیام

برای تعیین آیکن کادر پیام از ثابت‌های زیر استفاده کنید. فقط دقت کنید که هر یک از این آیکن‌ها را در جای مناسب به کار ببرید.

مقدار	نام ثابت	مفهوم	آیکن
16	vbCritical	آیکن پیام بحرانی	
32	vbQuestion	آیکن علامت سوال	
48	vbExclamation	آیکن هشدار یا افطار	
64	vbInformation	آیکن اطلاعات	

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مثال:

```
Response = MsgBox ("Is the printer on?", vbYesNoCancel + vbQuestion)
```

Or

```
Response = MsgBox ("Is the printer on?", 3 + 32)
```

نکته:

اگر به دو جدول بالا دقت کرده باشید ستونی تحت عنوان "نام ثابت" در آنها مشاهده می‌کنید. ویژگی‌های یسیک صدها ثابت (Constant) دارد که آنها را به نام می‌شناسد. به اینها ثابت نام‌دار یا (Named Constant) می‌گویند. مقدار این ثابت‌ها همیشه ثابت است (و غیر از این هم نمی‌تواند باشد) و نمی‌توان آنها را تغییر داد. وقتی از ثابت‌های نام‌دار استفاده می‌کنید، دیگر نیازی نیست مقدار آنها را حفظ کنید و دفعه‌ی بعد که آنها را ببینید، درک عملکرد آنها ساده‌تر خواهد بود. بنابراین سعی کنید تا حد امکان از ثابت‌های نام‌دار استفاده کنید. (طبق این تعاریف، استفاده از مثال اول مناسب‌تر است).

بخش سوم: تعیین دکمه‌ی پیش‌فرض

همیشه اولین دکمه‌ی کادر پیام (از سمت چپ) دکمه‌ی پیش‌فرض است، یعنی زدن Enter معادل کلیک کردن آن دکمه خواهد بود؛ اما با استفاده از مقادیر جدول زیر می‌توانید این رفتار را عوض کنید.

مقدار	نام ثابت	مفهوم
0	vbDefaultButton1	کلید اول پیش‌فرض
256	vbDefaultButton2	کلید دوم پیش‌فرض
512	vbDefaultButton3	کلید سوم پیش‌فرض
768	vbDefaultButton4	کلید چهارم پیش‌فرض

مثال:

```
Response = MsgBox ("Is the printer on?", vbYesNoCancel + vbDefaultButton1)
```

Or

```
Response = MsgBox ("Is the printer on?", 3 + 0)
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

در این مثال، دکمه‌ی Yes به عنوان دکمه‌ی پیش‌فرض در نظر گرفته شده است. در اعمال بهرانی، مانند حذف فایل‌ها، دکمه‌ی Cancel را دکمه‌ی پیش‌فرض کادر پیام قرار دهید تا اگر کاربر تصادفاً کلید Enter را فشار داد، عملیات لغو شود.

مثال‌های تکمیلی:

شماره‌ی یک:

```
Private Sub Form_Load()  
Dim intResponse As Integer  
intResponse = MsgBox("Exit Program?", vbInformation + vbYesNo, "Exit")  
If intResponse = vbYes Then  
    End  
Else  
    Me.Refresh  
End If  
End Sub
```

Or

```
Dim intResponse As VbMsgBoxResult  
intResponse = MsgBox("Exit Program?", 64 + 4, "Exit")  
If intResponse = 6 Then  
    End  
Else  
    Me.Refresh  
End If  
End Sub
```

نکته:

Me.Refresh باعث بارگذاری مجدد Me یعنی فرم جاری (Form1) می‌شود. بعدها با این دستور بیشتر آشنا می‌شوید.

شماره‌ی دو:

```
Option Explicit  
Dim p, s, t, r  
'p = Prompt  
's = Style  
't = Title  
'r = Response
```

```
Private Sub Command1_Click()
```

```
p = "Are You Sure?"
```

```
s = vbCritical + vbYesNo
```

```
t = "Yes or NO"
```

```
r = MsgBox(p, s, t)
```

```
End Sub
```

نکته:

اگر می‌خواهید در کادر پیام، بعد از نوشتن چندین کلمه به خط بعدی بروید و ادامه پیام را در آن خط بنویسید، باید از تابع **Chr\$(13)** استفاده کنید. این تابع کاراکتری را که مربوط به Charcode می‌باشد را برمی‌گرداند: (به جای **Chr\$(13)** می‌توانید از **Chr\$(10)** هم استفاده کنید)

Chr\$ (Charcode)

```
MsgBox ("Welcome" + Chr$(13) + Chr$(13) + Chr$(13) + "Meisam")
```

Or

```
MsgBox ("Welcome" + Chr(13) + Chr(13) + Chr(13) & "Meisam")
```

=====

تابع InputBox()

در بعضی از موارد ممکن است بخواهید برای کاربر سوالی را مطرح کنید و پاسخ آن را بگیرید، اما برای این منظور قرار دادن یک کادر متن (TextBox) بر روی فرم برنامه را مناسب نمی‌بینید. حال این سوال برای شما پیش می‌آید که آیا راه دیگری برای ارتباط با کاربر وجود دارد؟

بله. تابع **InputBox** یک کادر ورودی برای شما ایجاد می‌کند. این کادر تمام خصوصیات کادر پیام را دارد و فقط دارای یک فیلد اضافی برای گرفتن جواب کاربر است. البته برنامه‌نویس هیچ کنترلی روی دکمه‌های کادر ورودی ندارد و یک کادر ورودی همیشه دارای دو دکمه‌ی OK و Cancel خواهد بود. کادر ورودی، آیکن هم نمی‌تواند داشته باشد. شکل کلی تابع **InputBox()** به صورت زیر است:

```
varResponse = InputBox (strPrompt [, strTitle] [, strDefault] [, intXpos] [, intYpos])
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مقدار برگشتی تابع InputBox() از نوع Variant است و می‌توان آن را به صورت یک رشته به کار برد؛ به این دلیل مقدار برگشتی این تابع Variant است که بتوان آن را حتی به مشفیه‌های کنترل‌ها هم نسبت داد. از میان تمام آرگومان‌های تابع InputBox() فقط اولین آرگومان اجباری است.

- **strPrompt**: پیام یا پرسشی است که در کادر ورودی مشاهده می‌شود. حداکثر طول این پیام، می‌تواند 1024 نویسه باشد.

- **strTitle**: عنوان پنجره‌ی کادر ورودی با این آرگومان مشخص می‌شود.

- **strDefault**: مقداری که به صورت پیش‌فرض در فیلد ورودی ظاهر خواهد شد.

- **intXpos** و **intYpos**: مختصات ظاهر شدن پنجره‌ی کادر ورودی روی صفحه. اگر این دو آرگومان قید نشوند، ویژوال بیسیک کادر ورودی را وسط صفحه‌ی نمایش قرار خواهد داد.

نکته:

باید روشی وجود داشته باشد تا برنامه براند که کاربر کدام دکمه را کلیک کرده است. اگر کاربر روی Cancel کلیک کند، تابع InputBox() رشته‌ای به طول صفر ("") برمی‌گرداند و کلیک کردن روی OK سبب برگشت رشته‌ی دافل فیلد ورودی خواهد شد.

مثال‌ها:

کادر پیام، کادر ورودی

```
Private Sub Command1_Click()  
Dim strAnswer As String  
strAnswer = InputBox("Enter your name", "Getting name", "Meisam")  
MsgBox ("Hi " & strAnswer)  
End Sub
```

ضرب دو عدد

ضرب دو عدد

```
Private Sub Command1_Click()  
Dim intNum1, intNum2, intResult As Integer
```

```
intNum1 = InputBox("Enter First number", "Getting Number 1")
intNum2 = InputBox("Enter second number", "Getting Number 2")
intResult = intNum1 * intNum2
MsgBox intNum1 & " * " & intNum2 & " = " & intResult, "Result"
End Sub
```

دستور Select Case

بهترین روش برای بررسی شرطهای چندگانه، استفاده از دستور Select Case می باشد. اگر تعداد دستورهای If تودرتو سه یا چهار عدد بیشتر شود، برنامه بسیار پیچیده خواهد شد. به شکل کلی دستور Select Case توجه کنید:

عبارت مورد نظر Select Case

مقدار اول Case

یک یا چند دستور

[Case مقدار دوم

یک یا چند دستور

[Case مقدار سوم

یک یا چند دستور

...

[Case مقدار N ام

یک یا چند دستور

[Case Else

یک یا چند دستور

End Select

در این دستور عبارتی را که می خواهیم مقادیر مختلف آن را چک کنیم، در جلوی Select Case می نویسیم. اگر عبارت با یکی از مقادیر Case ها برابر باشد، دستورهای بعد از آن، و در غیر این صورت، دستورهای بعد از Case Else اجرا می شوند.

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

مثال‌ها:

نمرات و جوایز ۱

```
Private Sub cmdRun_Click()  
Select Case txtGrade.Text  
Case "a"  
    lblReward.Caption = "a Brand-New car"  
Case "b"  
    lblReward.Caption = "Notebook"  
Case "c"  
    lblReward.Caption = "Pencil"  
Case "d"  
    lblReward.Caption = "Eraser"  
Case "e"  
    lblReward.Caption = "Nothing"  
Case Else  
    lblReward.Caption = "Error in grade"  
End Select  
txtGrade.SetFocus  
End Sub
```

در مثال بالا اگر جعبه‌متن (txtGrade) حاوی یکی از حروف a, b, c, d و یا e باشد، دستور بعد از Case اجرا می‌شود؛ در غیر این صورت برنامۀ پیغام "Error in grade" را نمایش می‌دهد. در ضمن برنامۀ به بزرگی و کوچکی حروف حساس می‌باشد.

نمرات و جوایز ۲

برای استفاده از عملگرهای رابطۀ ای در دستور Select Case باید از عملگر Is استفاده کنید. برنامۀ زیر نوع دیگری از مثال نمرات و جوایز می‌باشد:

```
Private Sub cmdRun_Click()  
Select Case Val(txtGrade.Text)  
Case Is >= 18  
    lblReward.Caption = "a Brand-New car"  
Case Is >= 16  
    lblReward.Caption = "Notebook"
```

بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

```
Case Is >= 14
    lblReward.Caption = "Pencil"
Case Is >= 10
    lblReward.Caption = "Eraser"
Case Is >= 1
    lblReward.Caption = "Nothing"
Case Else
    lblReward.Caption = "Error in grade"
End Select
txtGrade.Text = ""
txtGrade.SetFocus
End Sub
```

کنجکاوی:

عبارات زیر را به عنوان ورودی به برنامه بدهید تا ببینید چه جایزه‌ای به شما می‌دهد:

1TM
11FS
426NB
NB426

فکر می‌کنید علت چیست؟

نمرات و پوایز ۳

برای ایجاد مصدوره در دستور Select Case باید از عملگر To استفاده کنید. این عملگر علاوه بر اعداد می‌تواند روی رشته‌ها نیز عمل کند، مشروط بر اینکه بررسی از پایین جدول ASCII شروع شود و به سمت بالا حرکت کند. برنامه‌ی زیر نوع دیگری از مثال نمرات و پوایز می‌باشد:

```
Private Sub cmdRun_Click()
Select Case Val(txtGrade.Text)
Case 18 To 20
    lblReward.Caption = "a Brand-New car"
Case 15 To 17
    lblReward.Caption = "Notebook"
Case 14 To 16
    lblReward.Caption = "Pencil"
```


Case 10 To 13

lblReward.Caption = "Eraser"

Case 0 To 9

lblReward.Caption = "Nothing"

Case Else

lblReward.Caption = "Error in grade"

End Select

txtGrade.Text = ""

txtGrade.SetFocus

End Sub

کنترل Check Box

از این کنترل که به عنوان کادر علامت شناخته می‌شود، برای انتقال یک حالت از دو حالت ممکن مورد استفاده قرار می‌گیرد. این کنترل تداعی‌گر متغیر منطقی می‌باشد. کنترل کادر علامت را می‌توان به صورت گروهی نیز به کار برد تا چندین انتقال را به کاربر ارائه دهد و کاربر می‌تواند یک یا چند کادر علامت را انتقال کند.

خاصیت Value:

مشخصه‌ی Value کنترل کادر علامت، تعیین می‌کند که آیا کادر علامت به وسیله‌ی کاربر انتقال شده است یا نه و آیا این کادر غیرفعال است؟ جدول زیر وضعیت کادر علامت، با مقدار مختلف را مشخص می‌کند:

وضعیت	مقدار	ثابت
Unchecked	0	vbUnchecked
Checked	1	vbChecked
Unavailable	2	vbGrayed

به طور پیش‌فرض کنترل کادر علامت با vbUnchecked مقداردهی می‌شود. در صورتی که می‌خواهید تعدادی از کادرهای علامت را از قبل انتقال کنید، می‌توانید مشخصه‌ی Value آنها را در

بهترین وبلاگ فارسی در ارائه ایبوک

» IR-E-BOOK.Blogfa.com «

روال‌های Form_Load، Form_Initialize، Form_Activate، vbChecked یا vbGrayed هم زمانی استفاده می‌شود که انتخاب ما شرط دارد است.

مثال‌ها:

تغییر ظاهری لیبل



```
Private Sub chkBold_Click()  
If chkBold.Value = 1 Then Label1.Font.Bold = True Else Label1.Font.Bold = False  
End Sub
```

```
Private Sub chkUnder_Click()  
If chkUnder.Value = 1 Then  
Label1.Font.Underline = True  
Else  
Label1.Font.Underline = False  
End if  
End Sub
```

```
Private Sub chkIta_Click()  
If chkIta.Value = 1 Then Label1.Font.Italic = True Else Label1.Font.Italic = False  
End Sub
```

```
Private Sub chkStrike_Click()  
If chkStrike.Value = 1 Then  
Label1.Font.Strikethrough = True  
Else  
Label1.Font.Strikethrough = False  
End if  
End Sub
```

```
Private Sub chkNormal_Click()  
chkBold.Value = 0
```

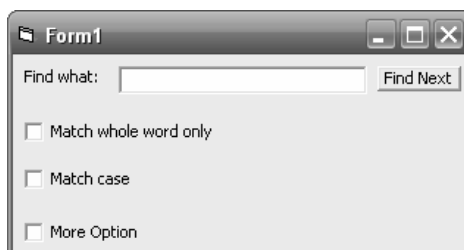
بهترین وبلاگ فارسی در ارائه ایبوک

IR-E-BOOK.Blogfa.com

```
chkUnder.Value = 0  
chkIta.Value = 0  
chkStrike.Value = 0  
chkNormal.Value = 0  
End Sub
```

گزینه‌های بیشتر

ابتدا ارتفاع فرم را زیاد می‌کنیم و کنترل‌های موردنظرمان را در بخش انتهایی فرم قرار می‌دهیم. حال می‌فواهیم با کلیک روی گزینه‌ی More Option گزینه‌های بیشتری را به نمایش درآوریم. برای این کار از تکه‌کد زیر استفاده کنید. این برنامه فقط جنبه‌ی شبیه‌سازی دارد.



```
Private Sub MoreOption_Click()  
If MoreOption.Value = 1 Then  
Form1.Height = Form1.Height + 780  
Else  
Form1.Height = 2520  
End If  
End Sub
```



عدد 2520 موجود در مقابل Form1.Height نشان‌دهنده‌ی ارتفاع فرم ما در زمان طراحی می‌باشد. حال برنامه را طوری تغییر دهید که بعد از کلیک کردن روی MoreOption این‌کادر علامت دیده نشود و گزینه‌های زیرین آن در زیر Match Case قرار بگیرند (با فاصله‌های یکسان). نکته:

بهترین وبلاگ فارسی در ارائه آیبوک

IR-E-BOOK.Blogfa.com

در مثال بالا و در نامگذاری کنترل کادر علامت از پیشنوندهای سه حرفی که یکپرده‌ای از نام کنترل می‌باشد استفاده کردیم، این پیشنوندها که از سوی مایکروسافت توصیه شده است، خوانایی برنامه را بالا می‌برد و از سردرگمی برنامه‌نویس جلوگیری می‌کند. جدول زیر این پیشنوندها را با مثالی توضیح داده است:

پیشوند	نام کنترل	مثال
pic	PictureBox	picMonitor
lbl	Label	lblOutPut
txt	TextBox	txtLastName
frm	Frame	frmEntry
cmd	CommandButton	cmdRun
chk	CheckBox	chkBold
opt	OptionButton	optAgreement
cbo	ComboBox	cboPersian
lst	ListBox	lstStudents
hsb	HScrollBar	hsbColor
vsb	VScrollBar	vsbSmall
tmr	Timer	tmrAlarm
drv	DriveListBox	drvTarget
dir	DirListBox	dirSource
fil	FileListBox	filSource
shp	Shape	shpCircle
lin	Line	linSeparator
img	Image	imgEEPROM

برخی دیگر از پیشنوندهای معمول:

پیشوند	نام کنترل	مثال
tlb	Toolbar	tbrStandard
tre	Tree view	treCodeHead

sta	Status bar	staDateTime
rtf	Rich text box	rtfReport
prg	Progress bar	prgLoadFile
mnu	Menu	mnuFile
ils	Image list	ilsIcons
dlg	Common dialog	dlgSave

کنترل Option Button

همانطور که گفته شد از دکمه‌ی **انتخاب** یا Option Button برای انتخاب یک گزینه از میان چند گزینه استفاده می‌شود. ویژوال بیسیک اجازه نمی‌دهد که در هر لحظه، بیش از یکی از دکمه‌ها انتخاب شود.

نکته:

هرگز یک دکمه‌ی انتخاب را به تنهایی روی یک فرم قرار ندهید، زیرا فعال کردن آن ممکن است، ولی دیگر امکان غیرفعال کردن آن وجود نخواهد داشت. دکمه‌های انتخاب فقط وقتی غیرفعال می‌شوند که کاربر روی دکمه‌ی دیگری کلیک کند.

مثال:

تعیین رنگ برپسب

```
Private Sub optR_Click()  
If optR.Value = True Then Label1.ForeColor = vbRed  
End Sub  
  
Private Sub optB_Click()  
If optB.Value = True Then Label1.ForeColor = vbBlue  
End Sub  
  
Private Sub optG_Click()  
If optG.Value = True Then Label1.ForeColor = vbGreen  
End Sub
```



کنترل Frame

در مثال بالا مشاهده کردید که در هر لحظه فقط یکی از دکمه‌های انتخاب را می‌توان انتخاب (فعال) کرد؛ حتی اگر تعداد دکمه‌های انتخاب موجود در یک فرم بیش از ۱۰۰ تا باشد باز فقط یکی از آنها را می‌توان انتخاب نمود، ولی از نظر تکنیکی می‌توان در یک لحظه، روی یک فرم چندین دکمه‌ی انتخاب فعال داشت. این کار با استفاده از کنترل فریم امکان‌پذیر است. همانطور که در فصل اول گفته شد وظیفه‌ی این کنترل دسته‌بندی ظاهری و منطقی کنترل‌های دیگر است. فریم که به آن ظرف یا Container هم می‌گویند، مکانی برای قرار گرفتن کنترل‌های دیگر است.

نکته:

به یاد داشته باشید که فریم باید قبل از دکمه‌های انتخاب روی فرم قرار داده شود.

مثال‌ها:

تعیین رنگ زمینه و رنگ قلم

فرمی به شکل زیر طراحی کنید!



حال برای آنکه ویژوال بیسیک بتواند متوجه شود که یک دکمه‌ی انتخاب باید داخل فریم قرار گیرد (نه روی فرم) بایستی آن را روی فریم رسم کنید. برای این کار روی `OptionButton` در جعبه ابزار

بهترین وبلاگ فارسی در ارائه ایبوک

» **IR-E-BOOK.Blogfa.com** «

کلیک کرده و سپس روی فریم، آن را به اندازه‌ی مورد نظر رسم کنید، یا ابتدا OptionButton را،
روی فرم قرار دهید و سپس با انجام Cut و Paste آن را درون فریم قرار دهید؛

```
Private Sub CmdExit_Click()  
    Unload Me  
End Sub
```

```
Private Sub Form_Load()  
    Option1.Value = False  
    Option2.Value = False  
    Option3.Value = False  
    Option4.Value = False  
    Option5.Value = False  
    Option6.Value = False  
End Sub
```

```
Private Sub Option1_Click()  
    If Option1.Value = True Then Label1.BackColor = vbBlue  
End Sub
```

```
Private Sub Option2_Click()  
    If Option2.Value = True Then Label1.BackColor = vbYellow  
End Sub
```

```
Private Sub Option3_Click()  
    If Option3.Value = True Then Label1.BackColor = vbWhite  
End Sub
```

```
Private Sub Option4_Click()  
    If Option4.Value = True Then Label1.ForeColor = vbRed  
End Sub
```

```
Private Sub Option5_Click()  
    If Option5.Value = True Then Label1.ForeColor = vbGreen  
End Sub
```


بهترین وبلاگ فارسی در ارائه ایبوک

>> **IR-E-BOOK.Blogfa.com** <<

Private Sub Option6_Click()

If Option6.Value = True Then Label1.ForeColor = vbCyan

End Sub

=====

پایان