آموزش گام به گام برنامه‌نویسی
بانک اطلاعاتی با ویژوال بیسیک‌نت
(مرجع کامل)

برخی از عناوین مهم

گزارشگیری Microsoft Report

پیاده سازی پرس و جوی پویا

پشتیبان گیری از بانک اطلاعاتی بصورت فشرده

ذخیره و بازیابی تصاویر در بانک اطلاعاتی

بیان کلاسهای متعدد ازقبیل

DataSet SQL Connection SQL Commands,...

تالیف:
مهندس رمضان عباس‌نژادورزی

آموزش گام به گام برنامه‌نویسی بانک اطلاعاتی با ویژوال بیسیک‌نت (مرجع کامل)

تالیف

مهندس رمضان عباس‌نژادورزی

فن‌آوری نوین

سرشناسه	:	عباس نژاد، رمضان، ۱۳٤۸ –
عنوان و نام پدیدآور	:	آموزش گام به گام برنامه‌نویسی بانک اطلاعاتی با ویژوال بیسیک‌نت(مرجع کامل)/تالیف رمضان عباس‌نژادورزی.
مشخصات نشر	:	بابل: فناوری نوین، ۱۳۸۸.
مشخصات ظاهری	:	۲۸۸ص.:مصور،جدول.
شابک	:	٦٦۰۰۰ریال:۹۷۸٦۰۰۹۱٤۱۳۵۷
وضعیت فهرست نویسی	:	فیپا
موضوع	:	ویژوال بیسیک مایکروسافت
موضوع	:	ویژوال بیسیک (زبان برنامه‌نویسی کامپیوتر)
موضوع	:	مایکروسافت دات نت
موضوع	:	پایگاه‌های اطلاعاتی – مدیریت
رده بندی کنگره	:	QA۷٦/۷۳/ب۹ع۱٦ ۱۳۸۸
رده بندی دیویی	:	۰۰۵/۲٦۸
شماره کتابشناسی ملی	:	۸٤۷۷۵۹۱

فن‌اوری نوین

www.fanavarinovin.net

بابل، کدپستی ۴۷۱۶۷-۷۳۴۴۸ تلفن: ۲۲۵۶۶۸۷-۰۱۱۱

آموزش گام به گام برنامه‌نویسی بانک اطلاعاتی با ویژوال بیسیک‌نت (مرجع کامل)

تألیف: مهندس رمضان عباس‌نژادورزی

ناشر: فن‌آوری نوین

چاپ اول: زمستان ۱۳۸۸

تیراژ :۱۰۰۰ جلد

طراح جلد: احمد فرجی

شابک: ۷- ۵- ۹۱۴۱۳ - ۶۰۰ - ۹۷۸

حروفچینی و صفحه‌آرایی: فن‌آوری نوین **قیمت: ۶۶۰۰ تومان**

تهران، خ اردیبهشت، نبش وحید نظری، پلاک ۱۴۲ **تلفکس: ۶۶۴۰۰۱۴۴-۶۶۴۰۰۲۲۰**

فهرست مطالب

مقدمه

امروزه حجم زیادی از اطلاعات ذخیره و بازیابی می‌شوند. برای جلوگیری از افزونگی داده (تکرار بی‌مورد داده‌ها)، بی نظمی و ایجاد سازگاری بین گزارش‌ها از بانک اطلاعات استفاده می‌شود. بانک‌های اطلاعات متعددی وجود دارند که پرکاربردترین و بهترین آنها، سیستم بانک اطلاعات رابطه‌ای است. یکی از بانک‌های اطلاعات رابطه‌ای، **SQL Server** است اکنون نسخه ۲۰۰۸ این بانک اطلاعاتی در بازار موجود است. اما، این کتاب طوری طراحی شده است که وابسته به نسخه خاصی از بانک اطلاعاتی نباشد. زیرا در این کتاب سعی شده است تمام کارها از طریق دستورات **SQL** انجام شود. ویژگی‌هایی از قبیل کارایی بالا، سهولت یادگیری و استفاده، کارکردن در محیط شبکه، قابلیت دسترسی و امنیت بالا، آن را به عنوان پرکاربردترین بانک اطلاعات جهان تبدیل کرده است. به طوری که ۷۰ درصد کاربران دنیا از این بانک اطلاعات استفاده می‌کنند.

از طرف دیگر، با بانک اطلاعات **SQLServer** نمی‌توان کارهایی از قبیل ایجاد فرم‌های ورود و ویرایش اطلاعات، تهیه گزارش‌ها و غیره را انجام داد. به همین دلیل، برای انجام کارهای مذکور به زبان برنامه‌نویسی نیاز داریم. از آنجایی زبان برنامه‌نویسی ویژوال بیسیک‌دات‌نت، یک زبان ساده و کار آمد است که افراد زیادی از آن بهره می‌گیرند. از نکات بارز این کتاب آموزش گزارش‌گیری **Microsoft Report** است که در فصل ۹ آن را می‌بینید. این زبان را به عنوان زبان برنامه‌نویسی بانک اطلاعات انتخاب نمودیم.

در این کتاب مطالبی از قبیل ایجاد بانک اطلاعات، جداول، کلاس‌های ویژوال بیسیک‌نت برای کار با بانک اطلاعات، ورود، ویرایش، حذف رکوردها، رویه‌های ذخیره شده، پشتیبان‌گیری و بازیابی فایل‌های پشتیبان به صورت فشرده شده، تراکنش‌ها و گزارش‌گیری از بانک اطلاعات بیان گردید.

کتاب حاضر با بهره‌گیری از سال‌ها تجربه در امر تدریس، تألیف کتب کامپیوتر و مهم‌تر از همه برنامه‌نویسی در زمینه بانک اطلاعات تدوین شده است. از ویژگی‌های جالب و برجسته این کتاب، بیان مثال‌های متنوع کاربردی، حل گام به گام آنها و توضیح کامل مثال‌های بیان شده، می‌باشد.

در پایان امیدوارم این اثر مورد توجه اساتید و دانشجویان عزیز واقع شود.

کتاب حاوی برنامه‌هایی است که کد آنها را می‌توانید به صورت رایگان از سایت انتشارات فن‌آوری نوین به آدرس **www.fanavarinovin.net** بگیرید.

رمضان عباس نژادورزی
fanavarienovin@yahoo.com

فصل ۱ آشنایی با زبان ویژوال بیسیک‌نت

سازمان‌ها برای نگهداری داده‌ها از بانک اطلاعاتی استفاده می‌کنند. یکی از پرکاربردترین نرم‌افزارهای مدیریت بانک‌های اطلاعات، SQL Server است. از طرف دیگر، بانک اطلاعات به تنهایی نمی‌تواند نیازهای سازمان‌ها را برطرف کند. به همین دلیل با یکی از زبان‌های برنامه‌سازی باید بتوان به بانک اطلاعات متصل شده، داده‌های آن را ویرایش، حذف و اضافه نمود. امروزه صدها زبان برنامه‌سازی وجود دارند که از طریق آنها می‌توان به بانک اطلاعات متصل گردید و داده‌های آن را دستکاری نمود. یکی از مهم‌ترین و پرکاربردترین زبان‌های برنامه‌سازی، ویژوال بیسیک نت می‌باشد. به همین دلیل در این فصل به طور خلاصه زبان ویژوال بیسیک‌نت را می‌آموزیم.

۱-۱. زبان ویژوال بیسیک نت

این زبان به همراه نرم‌افزار ویژوال استودیونت ارائه شده است. زبان ویژوال بیسیک‌نت از فناوری شیء[1] و مفهوم شی‌گرایی[2] استفاده می‌کند. هر چیزی که در دنیای واقعی وجود دارد، **شیء** نامیده می‌شود. مثل مردم، ساختمان‌ها، کارخانه‌ها، گیاهان، اتومبیل‌ها، کامپیوترها و غیره. هر شیء از **صفاتی** مانند اندازه، رنگ، وزن و دیگر صفات تشکیل شده است که شکل ظاهری آن را تعیین می‌کنند و رفتارهایی از خودشان نشان می‌دهند، مانند انسان می‌خوابد، گریه می‌کند، می‌خندد، اتومبیل حرکت می‌کند، ترمز می‌نماید و غیره.

از آنجایی که زبان ویژوال بیسیک نت از فناوری شی‌گرایی استفاده می‌نماید، امکاناتی در این زبان اضافه شده است که می‌توانید اشیای دنیای واقعی را مدل‌سازی کنید. برای این که ویژوال بیسیک نت شی‌گرایی را پیاده‌سازی کند از مفهوم **کلاس**[3] استفاده می‌کند. **کلاس**، برای ایجاد انواع جدید به کار می‌رود. هر کلاس شامل داده‌ها و متدهایی است که داده‌ها را دستکاری می‌کنند و سرویس‌هایی را برای مشتریان فراهم می‌نمایند. با مفهوم کلاس و چگونگی پیاده‌سازی آنها در ادامه بیشتر آشنا خواهیم شد.

۲-۱. فضای نام

همان‌طور که بیان گردید، زبان برنامه‌نویسی ویژوال بیسیک نت از کلاس برای برنامه‌نویسی استفاده می‌کند. کلاس‌ها به دو دسته تقسیم می‌شوند که عبارتند از:

۱. **کلاس‌های آماده**: این کلاس‌ها از قبل نوشته شده‌اند و در کتابخانه FCL ویژوال استودیونت وجود دارند.

---

[1]- Object [2]- Object Oriented [3]- Class

۲. کلاس‌هایی که برنامه‌نویس می‌نویسد: همه کلاس‌های مورد نیاز برنامه‌نویسان از قبل وجود ندارند. بنابراین برنامه‌نویس نیاز دارد، برخی از کلاس‌ها را بنویسد. در ادامه با این کلاس‌ها آشنا خواهید شد.

کلاس‌هایی که در کتابخانه FCL وجود دارند، در **فضاهای نام**[1] مختلف قرار می‌گیرند. برخی از فضای نام‌ها و وظایف آنها در جدول ۱-۱ آمده است. این فضاهای نام به طور خودکار به برنامه ویژوال بیسیک نت اضافه می‌شوند. علاوه بر این فضاهای نام، فضاهای نام دیگری وجود دارند که می‌توانید آنها را به برنامه اضافه کنید. برای این منظور باید از دستور Imports استفاده نمایید. به عنوان مثال، دستورات زیر را ببینید:

```
Imports System.Data.SqlClient
Imports System.Convert
```

دستور اول، فضای نام System.Data.SqlClient را به برنامه اضافه می‌کند تا بتوانید از کلاس‌هایی که برای اتصال و دستکاری به بانک اطلاعات SQL Server به کارمی‌روند، بهره بگیرید (با این فضای نام در ادامه بیشتر آشنا خواهید شد) و دستور دوم، فضای نام System.Convert را به برنامه اضافه می‌کند تا بتوانید از متدهایی که برای تبدیل انواع داده‌های مختلف به کار می‌روند، استفاده نمایید.

جدول ۱-۱ برخی از فضاهای نام موجود در FCL.

فضای نام	هدف
System	حاوی کلاس‌های پایه و انواع داده از قبیل double، int، char و غیره است.
System.Data	دارای کلاس‌هایی است که برای دستیابی به داده‌های بانک اطلاعات استفاده می‌شوند.
System.Io	از کلاس‌هایی تشکیل شده است که برای ورودی- خروجی داده‌ها مانند فایل‌ها به کار می‌روند.
System.Drawing	از کلاس‌هایی تشکیل شده است که برای ترسیم اشکال گرافیکی به کار می‌رود.
System.Linq	از کلاس‌هایی تشکیل شده است که برای کار با LINQ به کار می‌روند.

۳-۱. انواع داده‌ها

اکثر برنامه‌ها با دریافت داده‌ها، پردازش و استخراج نتایج سر و کار دارند. یعنی، داده‌ها مهم‌ترین بخش برنامه‌نویسی را ایفا می‌نمایند. لذا، باید انواع داده‌هایی که در زبان برنامه‌نویسی وجود دارند، را بیاموزیم. دو نوع داده را می‌توان در زبان ویژوال بیسیک نت تعریف نمود که عبارتند از:

۱. داده‌های مقدار ۲. داده‌های مرجع

داده‌های مقدار، همان داده‌هایی هستند که توسط انواع اولیه تعریف می‌شوند (بجز داده‌های Object و String). این انواع در جدول ۲-۱ آمده‌اند و **داده‌های مرجع**، تمام انواعی هستند که توسط برنامه‌نویس تعریف می‌شوند یا کلاس‌های هستند که از قبل تعریف شده‌اند. در ادامه پیاده‌سازی کلاس را خواهید دید.

[1]- Namespaces

جدول۲-۱ انواع داده‌های اولیه در ویژوال بیسیک نت.			
نوع در زبان ویژوال بیسیک نت	معادل.NET	اندازه	نگهداری می‌کند.
Byte	Byte	۱	مقادیر بدون علامت (۰ تا ۲۵۵)
Char	Char	۱	کارکترهای یونیکد۱۶ بیتی
Boolean	Boolean	۱	مقادیر True یا False
Decimal	Decimal	۸	نقطه اعشار ثابت تا ۲۸ رقم
Short	Int16	۲	مقادیر صحیح از ۳۲۷۶۸- تا ۳۲۷۶۷
Currency	Uint16	۸	برای محاسبات پولی
Integer	Int32	٤	مقادیر صحیح بین $۲^{۳۱}$- تا ($۲^{۳۱}$ - ۱)
Long	Int64	۸	مقادیر صحیح $۲^{۶۳}$- تا ($۲^{۶۳}$ - ۱)
Single	Single	٤	اعداد اعشاری $۱/۵\times۱۰^{-۴۵}$± تا $۳/۴\times۱۰^{۳۸}$± با ۷ رقم اعشار
Double	Double	۸	اعداد اعشاری $۵/۰\times۱۰^{-۳۲۴}$± تا $۱/۷\times۱۰^{۳۰۸}$± با۱۵تا۱۶رقم بعد از اعشار
String	String	-	رشته‌ای با طول متغیر از صفر تا ۲ میلیارد کاراکتر
Date	Date	۸	مقادیر ۰ تا ($۲^{۶۴}$ - ۱)
Object	Object		کلاس پایه تمام اشیا دات نت

۴-۱. متغیرها

متغیرها نامی برای کلمات حافظه هستند که دارای ویژگی‌های زیر می‌باشند:

- داده‌ها در آنها ذخیره می‌شوند.
- مقدار آنها در طول اجرای برنامه ممکن است تغییر کند.
- در یک لحظه خاص فقط یک مقدار را دارند.

برای استفاده از متغیرها باید نام، نوع و مقدار آنها را تعیین کرد.

۱-۴-۱. نامگذاری متغیرها

برای نامگذاری متغیرها در ویژوال بیسیک‌نت می‌توان از ترکیبی از حروف، ارقام و خط ربط (_) استفاده کرد. به طوری که اولین آنها رقم نباشد. به عنوان مثال، sum، count1، i_1 می‌توانند نام‌هایی برای متغیرها باشند، ولی 1count و hioh!book نمی‌توانند نام متغیرها باشند. زیرا، اولین نام با رقم 1 شروع شد و در دومین نام کارکتر ! استفاده گردید که کارکترهای مجاز نمی‌باشند.

۲-۴-۱. اعلان متغیرها

بعد از این که متغیرها را نامگذاری کردید باید نوع آنها را تعیین نمایید. یعنی، باید تعیین کنید متغیر چه نوع داده‌ای را ذخیره می‌کند. متغیرها به صورت زیر اعلان می‌گردند:

Dim نام متغیر As نوع داده

در این ساختار Dim و As کلمه کلیدی می‌باشند که برای تعریف متغیر به کار می روند. نوع داده را در جدول ۲-۱ دیدید.

اکنون دستورات زیر را ببینید:

```
Dim x As Integer
Dim  d As Double
Dim s1, s2 As String
Dim obj As Object
Dim y As Boolean
```

دستور اول، متغیر x را از نوع صحیح تعریف می‌کند. دستور دوم، متغیر d را از نوع double و دستور سوم، متغیرهای s1 و s2 را از نوع رشته‌ای تعریف می‌نماید. دستور چهارم، obj را از نوع Object تعریف می‌کند و دستور پنجم y را از نوع Boolean معرفی می‌نماید.

۳-۴-۱. مقدار دادن به متغیرها

برای مقداردهی به متغیرها روش‌های متعددی وجود دارد که عبارتند از:

۱. مقداردهی در زمان اعلان متغیر ۲. پس از تعریف نوع متغیر و با دستور انتساب (=)

۳. دستورات ورودی

به عنوان مثال، دستورات زیرا را ببینید:

```
Dim x As Integer = 15
Dim s As string = "good"
```

دستور اول، متغیر x را با مقدار 15 از نوع Integer تعریف می‌کند و دستور دوم، متغیر رشته‌ای s را از نوع String تعریف می‌نماید و رشته "good" را به آن تخصیص می‌دهد. اکنون دستورات زیر را ببینید:

```
Dim x As Integer
Dim  s As String
x = 10
s = "ویژوال بیسیک نت"
```

دستورات اول و دوم متغیرهای x و s را به ترتیب از نوع Integer و String تعریف می‌کنند. دستور سوم، مقدار متغیر x را برابر ۱۰ قرار می‌دهد و دستور چهارم، مقدار متغیر s را رشته ویژوال بیسیک‌نت تعیین می‌کند.

در ادامه با چگونگی مقداردهی متغیرها با دستورات ورودی آشنا خواهید شد.

۵-۱. ثوابت

ثوابت، مقادیری هستند که در برنامه وجود دارند ولی قابل تغییر نیستند. برای تعریف ثوابت از واژه const به صورت زیر استفاده می‌شود:

const نام = مقدار ثابت

به عنوان مثال، دستورات زیر را ببینید:

```
const PI = 3.14
const ch = '+'
```

دستور اول، ثابت PI را با مقدار 3.14 معرفی می‌کند و دستور دوم، ثابت ch را با مقدار '+' تعریف می‌نماید.

۶-۱. عملگرها

عملگرها، نمادهایی هستند که اعمال خاصی را انجام می‌دهند. به عنوان مثال، نماد '-' عملگری است که دو مقدار را از یکدیگر تفریق می‌کند. عملگرها در ویژوال بیسیک نت به انواع مختلف تقسیم می‌شوند که در زیر آمده‌اند:

- **عملگرهای محاسباتی**، عملگرهایی هستند که عمل محاسبات را بر روی عملوند انجام می‌دهند. این عملگرها در جدول ۳-۱ آمده‌اند.
- **عملگرهای رابطه‌ای**، عملگرهایی هستند که دو عملوند را با یکدیگر مقایسه می‌کنند (جدول ٤-۱ را ببینید).
- **عملگرهای منطقی**، عملگرهایی هستند که بر روی عبارت منطقی (True یا False) عمل می‌کنند (جدول‌های ۵-۱ و ٦-۱ را مشاهده کنید). حاصل عملگر AND، وقتی درست است که هر دو عملوند درست باشند. حاصل عملگر OR، وقتی درست است که حداقل یکی از عملوندها درست باشند. حاصل عملگر XOR ، وقتی درست است که فقط یکی از عملوندها درست باشند. حاصل عملگر EQV، وقتی درست است که هر دو عملوند یکسان باشند. حاصل عملگر IMP، وقتی درست است که دومین عملوندها درست باشند.
- **عملگرهای ترکیبی**، از ترکیب عملگرهای محاسباتی و علامت = ایجاد می‌شوند (جدول۷-۱).

جدول ۳-۱ عملگرهای محاسباتی.

عملگر	نام	x	y	مثال	نتیجه
-	تفریق و منهای یکانی	10	12	y-x -y	2 -12
+	جمع	17	5	x +y	22
*	ضرب	10	2	x*y	20
/	تقسیم	10	5	x/y	2
\	خارج قسمت تقسیم صحیح	10	4	x \ y	2
Mod	باقیمانده تقسیم صحیح	10	3	x Mod y	1
^	توان	2	10	x ^ y	1024

جدول ۴-۱ عملگرهای رابطه‌ای.

عملگر	نام	x	y	مثال	نتیجه
>	بزرگتر	10	12	x > y	False
>=	بزرگتر یا مساوی	10	7	x > = y	True
<	کوچکتر	10	7	x < y	False
<=	کوچکتر یا مساوی	10	12	x <= y	True
=	تساوی	10	17	x = y	False
<>	نامساوی	10	17	x <> y	True

جدول ۵-۱ عملکرد عملگرهای منطقی.

X	Y	NOT X	X AND Y	X OR Y	X XOR Y	X EQV Y
False	False	True	False	False	False	True
True	False	False	False	True	True	False
False	True	True	False	True	True	False
True	True	False	True	True	False	True

جدول ۶-۱ عملگرهای منطقی.

عملگر	نام	x	y	مثال	نتیجه
NOT	نقیض (not)	True	False	NOT x	False
AND	و (and)	10	12	x < 12 AND y > 10	True
OR	یا (or)	12	8	x > 12 OR y < 6	False
XOR	یا انحصاری	12	8	x > 12 XOR y < 6	False
EQV	هم ارزی	12	8	x > 12 EQV y < 6	True
IMP	هم ارزی	12	8	x > 12 IMP y < 6	False

جدول ۷-۱ عملگرهای ترکیبی.

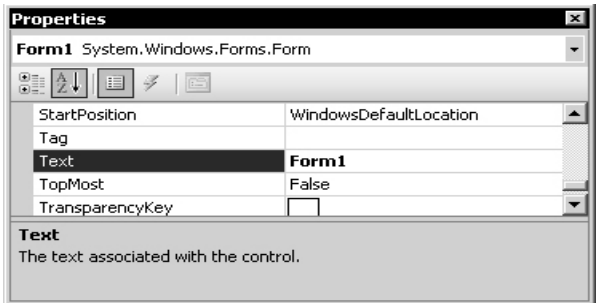
عملگر	نام	x	y	مثال	معادل	نتیجه
+ =	انتساب جمع	10	12	x += y	x = x + y	x = 22
- =	انتساب تفریق	10	8	x -= y	x = x - y	x = 2
* =	انتساب ضرب	10	12	x *= y	x = x * y	x = 120
/ =	انتساب تقسیم	10	2	x /= y	x = x / y	x = 5
^ =	انتساب توان	10	4	x ^= y	x = x ^ y	x = 10000

۷–۱. فرم برنامه

فرم برنامه، مکانی است که کنترل‌های برنامه در آن قرار می‌گیرند. هر برنامه بانک اطلاعات در ویژوال بیسیک‌نت حداقل یک فرم دارد. برای استفاده از فرم باید خواص، رویدادها و متدهای آن را بشناسیم. لذا در ادامه به این موضوعات می‌پردازیم.

۱–۷–۱. خواص فرم

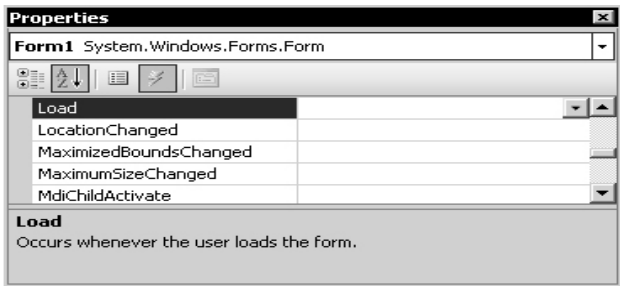
خواص فرم، شکل ظاهری فرم را تعیین می‌کنند. فرم خواص متعددی دارد. بیان همه این خواص از حوصله این کتاب خارج است. لذا، به تشریح خواص مهم فرم و کنترل‌های دیگر می‌پردازیم. برخی از خواص مهم فرم در جدول ۸–۱ آمده است. برای نمایش خواص فرم ، بر روی آن کلیک کنید و سپس کلید F4 را بزنید تا خواص فرم را ببینید (شکل زیر):



۲–۷–۱. رویدادهای فرم

همان‌طور که بیان کردیم، برنامه‌های ویژوال منتظر رخ دادن رویدادهایی هستند که توسط کاربر انتخاب می‌شوند تا به آنها پاسخ دهند. یعنی، اگر برنامه پاسخ‌گو به رویدادی نوشته شده باشد، در صورت انتخاب آن رویداد توسط کاربر، این برنامه اجرا می‌شود. فرم دارای رویدادهای مختلفی است. برخی از مهم‌ترین آنها در جدول ۹–۱ آمده‌اند.

برای مشاهده رویدادهای فرم، بر روی آن کلیک کنید تا فرم انتخاب شود. اکنون گزینه View/ Properties Window را اجرا نمایید تا پنجره Properties ظاهر شود. در این پنجره دکمه Events را کلیک کنید تا لیست رویدادهای فرم را ببینید (شکل زیر):



جدول ۸-۱ خواص مهم فرم.

خاصیت	هدف
Name	نام فرم را تعیین می‌کند. برای اولین فرم، نام فرم Form1 است که می‌توانید آن را تغییر دهید.
ActiveForm	فرم فعال فعلی را تعیین می‌کند.
Enabled	تعیین‌می‌کند آیا فرم فعال است یا خیر. اگر فرم غیرفعال گردد، به‌هیچ‌رویدادی‌پاسخ نمی‌دهد.
Font	فونت فرم را تعیین می‌کند.
ForeColor	رنگ‌نوشته‌های‌روی‌فرم را تعیین می‌کند.
Language	زبان مورد استفاده در فرم را تعیین می‌کند.
MinimizeBox	تعیین می‌کند آیا دکمه کمینه در عنوان فرم ظاهر شود یا خیر.
RightToLeft	جهت‌نمایش‌اطلاعات‌را تعیین‌می‌کند(این خاصیت برای زبان فارسی مفید است).
Size	اندازه فرم را تعیین می‌کند.
Text	متنی را تعیین می‌کند که باید در عنوان فرم نمایش داده شود.
Location	محل قرار گرفتن فرم را تعیین می‌کند.
Opacity	سطح شفافیت فرم را تعیین می‌کند.
MaximumBox	تعیین می‌کند آیا دکمه بیشینه نمایش داده شود یا خیر.
MinimumBox	تعیین می‌کند آیا دکمه کمینه نمایش داده شود یا خیر.

جدول ۹-۱ رویدادهای مهم فرم.

رویداد	هدف
Activated	وقتی که فرم فعال می‌شود، رخ می‌دهد.
Click	وقتی رخ می‌دهد که فرم کلیک گردد.
FormClosed	وقتی رخ می‌دهد که فرم بسته شود.
FormClosing	وقتی رخ می‌دهد که فرم در حال بسته شدن باشد. این رویداد قبل از رویداد FormClosed رخ می‌دهد.
Deactivate	وقتی رخ می‌دهد که فرم غیرفعال گردد.
DoubleClick	وقتی که فرم کلیک مضاعف شود، رخ می‌دهد.
Enter	وقتی مکان‌نما وارد فرم می‌شود، رخ می‌دهد.
KeyDown	وقتی کلیدی فشرده شده، پایین می‌رود رخ می‌دهد.
KeyPress	وقتی کلیدی فشرده می‌شود، رخ می‌دهد. این رویداد قبل از رویداد KeyDown اتفاق می‌افتد.
KeyUP	وقتی کلید فشرده شده رها می‌گردد، رخ می‌دهد.
Leave	وقتی مکان‌نما فرم را ترک می‌کند رخ می‌دهد.

ادامه جدول ۹-۱ رویدادهای مهم فرم.	
رویداد	هدف
Load	وقتی فرمی باز می‌شود، رخ می‌دهد (قبل از نمایش فرم).
MouseDown	وقتی که کلید ماوس فشرده شود، رخ می‌دهد.
MouseEnter	وقتی مکان‌نمای ماوس وارد فرم شود، رخ می‌دهد.
MouseLeave	وقتی مکان‌نمای ماوس فرم را ترک می‌کند، رخ می‌دهد.
MouseUP	وقتی کلید فشرده شده ماوس رها می‌شود، رخ می‌دهد.
Move	وقتی فرم شروع به حرکت می‌کند، رخ می‌دهد.
Resize	وقتی اندازه فرم تغییر می‌یابد، رخ می‌دهد.
TextChanged	وقتی رخ می‌دهد که متن فرم (کنترل) یا خاصیت Text تغییر کند.
Shown	وقتی که فرم نمایش داده شود، رخ می‌دهد.
SizeChanged	وقتی رخ می‌دهد که مقدار خاصیت Size تغییر یابد.
MouseMove	وقتی رخ می‌دهد که ماوس روی فرم حرکت می‌کند.

۳-۷-۱. متدهای فرم

متدها، کارهای خاصی را بر روی فرم انجام می‌دهند. برخی از متدهای مهم فرم در جدول ۱۰-۱ آمده‌اند.

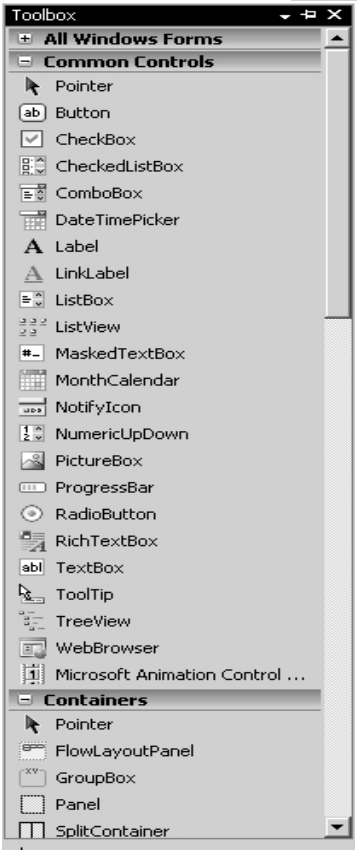
جدول ۱۰-۱ متدهای مهم فرم.	
متد	هدف
Active	فرم را فعال می‌کند.
Close	فرم را می‌بندد.
Dispose	فرم را حذف کرده از بین می‌برد.
Focus	مکان‌نما را به فرم مورد نظر منتقل می‌کند.
Hide	فرم را پنهان می‌کند.
ResetText	متن عنوان فرم را پاک می‌کند.
Show	فرم مخفی شده را آشکار می‌کند.
Refresh	فرم را بازسازی کرده، اطلاعات آن را دوباره رسم می‌کند.
ToString	محتویات فرم را به رشته تبدیل می‌نماید.
Validate	موجب اعتبارسنجی فرم می‌شود.

۸-۱. کنترل‌ها

همان‌طور که می‌دانید برای نوشتن برنامه‌ها در ویژوال بیسیک‌نت باید کنترل‌هایی را بر روی فرم قرار دهید (این کنترل‌ها همان قطعات تشکیل دهنده برنامه هستند). خواص آنها را مقداردهی کرده، برنامه پاسخ‌گو به

رویدادهای آنها را بنویسید. بنابراین، کنترل‌ها اجزا اصلی برنامه‌های ویژوال بیسیک‌نت را تشکیل می‌دهند. کنترل‌های زیادی در ویژوال بیسیک‌نت وجود دارند. بحث در مورد همه اینها در این کتاب نمی‌گنجد. لذا، در این کتاب برخی از کنترل‌های مهم را بررسی می‌کنیم.[۱]

این کنترل ما را در ادامه خواهید دید. برای اضافه کردن کنترل جدید بر روی فرم، دکمه Toolbox () را کلیک کنید تا کنترل‌های آماده را مشاهده کنید. اکنون جعبه ابزار ظاهر می‌شود (شکل ۱-۱). در این جعبه ابزار، کنترل مورد نظر را کلیک مضاعف کنید تا به فرم اضافه گردد و آن را به مکان دلخواه از فرم انتقال دهید (با کشیدن و رها کردن). من دکمه button1 را به فرم اضافه کردم و به مکان دلخواه انتقال دادم (شکل زیر):



شکل ۱-۱ پنجره ابزار ویژوال استودیونت

برای حذف کنترلی از فرم، آن را کلیک کرده تا انتخاب شود. اکنون دکمه Delete را فشار دهید تا کنترل از روی فرم حذف گردد.

۱-۸-۱. کنترل Label ()

این کنترل برای نمایش متن‌های غیرقابل ویرایش از قبیل عنوان فیلد و پیام‌های مورد نیاز به کار می‌رود. این کنترل نیز دارای خواص و رویدادهای زیادی است که برخی از مهم‌ترین آنها در زیر آمده است:

- **خاصیت AutoSize**: تعیین می‌کند آیا اندازه کنترل با توجه به مقدار خاصیت Text به طور خودکار تغییر کند یا خیر.
- **خاصیت TextAlign**: مکان قرار گرفتن متن در کنترل Label را تعیین می‌کند. به عنوان مثال، دستور زیر را ببینید.

```
label1.TextAlign = ContextAlignMent.TopLeft
```

این دستور، متن را در بالا سمت چپ نمایش می‌دهد.

- **رویداد AutoSizeChanged**: وقتی رخ می‌دهد که مقدار خاصیت AutoSize تغییر کند.
- **رویداد TextAlignChanged**: وقتی رخ می‌دهد که مقدار خاصیت TextAlign تغییر یابد.

---

[۱]- برای کسب اطلاعات بیشتر در مورد ویژوال بیسیک‌نت به کتاب آموزش گام به گام ویژوال بیسیک‌نت از انتشارات علوم رایانه، تألیف عین‌الله جعفرنژادقمی و رمضان عباس‌نژاد مراجعه کنید.

جدول ۱۱-۱ خواص، رویدادها و متدهای کنترل TextBox.

خاصیت	هدف
Lines	رشته‌ای از نوع آرایه است که تعداد خط‌های TextBox را تعیین می‌کند.
MaxLength	حداکثر طول متن را تعیین می‌کند. به عنوان مثال، اگر در این خاصیت ۵۰ را وارد کنید، کاربر حداکثر می‌تواند ۵۰ کاراکتر را وارد کند.
MultiLines	تعیین می‌کند آیا TextBox می‌تواند چند سطر اطلاعات را بپذیرد یا خیر.
Readonly	تعیین می‌کند که آیا محتویات TextBox فقط خواندنی باشد یا خیر. اگر اطلاعات TextBox فقط خواندنی باشد، این کنترل مانند Label عمل می‌کند.
PasswordChar	تعیین می‌کند در هنگام ورود اطلاعات در TextBox آیا اطلاعات به صورت کلمه عبور ظاهر شوند (اطلاعات اصلی مخفی گردند یا خیر).
رویداد	**هدف**
MultiLinesChanged	زمانی رخ می‌دهد که خاصیت MultiLines تغییر یابد.
ReadonlyChanged	زمانی رخ می‌دهد که خاصیت Readonly تغییر یابد.
متد	**هدف**
Clear	محتویات TextBox را پاک می‌کند.
ClearUndo	اثر اجرای فرمان Clear انجام شده را خنثی می‌کند.
Copy	متن انتخاب شده در TextBox را در حافظه موقت کپی می‌کند.
Cut	متن انتخاب شده در TextBox را به حافظه موقت انتقال می‌دهد.
Paste	متن حافظه موقت را در مکان فعلی کپی می‌نماید.
Undo	تأثیر آخرین فرمان انجام شده را خنثی می‌کند.
Select	برای انتخاب متن TextBox به کار می‌رود.
SelectAll	کل متن TextBox را انتخاب می‌کند.
DeselectAll	کلیه متن انتخاب شده TextBox را از حالت انتخاب خارج می‌کند.

۲-۸-۱. کنترل TextBox (abl TextBox)

این کنترل برای دریافت اطلاعاتی از قبیل مقادیر رشته‌ای، عددی و Memo (اطلاعات رشته‌ای طولانی) فیلدها به کار می‌رود. این کنترل نیز مانند کنترل‌های دیگر دارای خواص، رویدادها و متدهای متعددی است. برخی از خواص، رویدادها و متدهای این کنترل در جدول ۱۱-۱ آمده‌اند.

۳-۸-۱. کنترل Button (ab Button)

این کنترل‌ها به دکمه فرمان معروف هستند. زیرا با کلیک آن فرمان خاصی (دستورات خاصی) اجرا خواهد شد. خواص، رویدادها و متدهای این کنترل مانند کنترل‌های دیگر است. در ادامه بیشتر با این کنترل آشنا خواهید شد.

۴-۸-۱. کنترل ListBox (ListBox)

این کنترل برای نمایش لیستی از اشیاء به کار می‌رود. در این کنترل می‌توان اطلاعاتی از قبیل نام افراد، لیست کالاها و غیره را نمایش داد. این کنترل نیز مانند کنترل‌های دیگر دارای خواص، رویدادها و متدهای زیادی است که برخی از پرکاربردترین آنها در جدول ۱۲-۱ آمده‌اند.

جدول ۱۲-۱ خواص، متدها و رویدادهای کنترل ListBox.

خاصیت	هدف
Items	گزینه‌هایی را تعیین می‌کند تا بر روی کنترل نمایش داده شوند.
MultiColumn	تعیین می‌کند آیا کنترل ListBox می‌تواند چند ستونی باشد یا خیر.
SelectionMode	تعیین می‌کند آیا در کنترل ListBox می‌توان چند گزینه را انتخاب کرد. مقادیر این خاصیت می‌تواند None (انتخاب گزینه امکان‌پذیر نیست)، One (فقط یک گزینه را می‌توان انتخاب کرد)، MultiSimple (چند گزینه را می‌توان انتخاب کرد) و MultiExtended (با کلیدهای Shift و Ctrl می‌توان چند گزینه را انتخاب کرد) را بپذیرد.
DataSource	نام منبع داده (بانک اطلاعات) را تعیین می‌کند.
ValueMember	رشته‌ای را تعیین می‌نماید که باید در خاصیت DataSource نمایش داده شود.
Sorted	تعیین می‌کند آیا اطلاعات ListBox مرتب شده باشد یا خیر.
SelectedItems	لیستی است که گزینه‌های انتخاب شده را نگهداری می‌کند.
رویداد	**هدف**
DataSourceChanged	وقتی خاصیت DataSource تغییر یابد، رخ می‌دهد.
DisplayMemberChanged	وقتی خاصیت DisplayMember تغییر کند، رخ می‌دهد.
ValueMemberChanged	وقتی خاصیت ValueMember تغییر یابد، رخ می‌دهد.
متد	**هدف**
Add	برای اضافه کردن گزینه‌ای به انتهای ListBox به کار می‌رود.
Insert	گزینه‌ای را در مکان خاص ListBox اضافه می‌کند.
Count	تعداد گزینه‌های ListBox را می‌شمارد.
Remove	مقداری را از ListBox حذف می‌کند.
RemoveAt	مقداری را از مکان خاصی از ListBox حذف می‌کند.
Indexof	مقداری را دریافت کرده اندیس مکان آن را برمی‌گرداند.

اکنون دستورات زیر را ببینید:

```
listBox1.Clear()
listBox1.Sorted = true
listBox1.Items.Add ("one")
listBox1.Items.Insert ("zero" , 0)
listBox1.Items.RemoveAt (1)
```

دستور اول، گزینه‌های listBox1 را حذف می‌کند. دستور دوم، گزینه‌های listBox1 را به صورت مرتب شده نمایش می‌دهد. دستور سوم، گزینه one را به listBox1 اضافه می‌کند. دستور چهارم، گزینه "zero" را قبل از گزینه "one" اضافه می‌نماید و دستور پنجم، مقدار مکان دوم listBox1 را حذف می‌کند.

۵-۸-۱. کنترل ComboBox

این کنترل ترکیبی از یک کنترل TextBox و ListBox است که برای تایپ یا انتخاب گزینه‌ای به کار می‌رود. خواص، رویدادها و متدهای این کنترل مانند کنترل ListBox است.

۶-۸-۱. کنترل CheckBox (CheckBox)

این کنترل برای تعریف گزینه‌هایی با دو انتخاب از قبیل مرد یا زن ، جانباز یا غیرجانباز ، متأهل یا مجرد به کار می‌رود. این کنترل را می‌توان به فیلدهای منطقی در بانک اطلاعات انقیاد[1] کرد. در ادامه چگونگی انقیاد این کنترل را به فیلدهای منطقی می‌بینید. خواص و رویدادهای کنترل CheckBox در زیر آمده‌اند:

- **خاصیت CheckAlign**: محل قرار گرفتن ✓ (تیک) را در کنار مربع تعیین می‌کند.
- **خاصیت Checked**: تعیین می‌کند آیا کنترل CheckBox انتخاب شده است یا خیر (✓ نمایش داده شده است یا خیر).
- **خاصیت CheckState**: یکی از سه حالت CheckBox را تعیین می‌کند که می‌تواند مقادیر Unchecked (انتخاب نشده)، Checked (انتخاب شده) یا Indeterminate (غیرفعال) را بپذیرد.
- **خاصیت TnreeState**: تعیین می‌کند آیا کنترل CheckBox دارای سه حالت است یا خیر.
- **رویداد CheckedChanged**: وقتی رخ می‌دهد که خاصیت Checked تغییر یابد.
- **رویداد CheckStateChanged**: وقتی رخ می‌دهد که خاصیت CheckState تغییر یابد.

۷-۸-۱. کنترل CheckedListBox (CheckedListBox)

این کنترل ترکیبی از CheckBox و ListBox است. یعنی هر یک از گزینه‌های ListBox دارای حالت انتخاب CheckBox می‌باشند. خواص و رویدادهای این کنترل مانند رویدادهای CheckBox و ListBox است. ولی دو متد زیر به این کنترل اضافه شده است.

- **متد SetItemsChecked**: برای مقداردهی به خاصیت Checked این کنترل به کار می‌رود. به عنوان مثال، دستور زیر را ببینید.

```
CheckedListBox1.SetItemChecked (3, true)
```

این دستورحالت چهارمین گزینه انتخاب شده CheckListBox1 را تیک‌دار می‌کند (اندیس گزینه‌ها از صفر شروع می‌شود).

- **متد SetItemsCheckState**: مقدار خاصیت CheckState گزینه خاصی را تعیین می‌کند.

---

[1]- Bind

۸-۸-۱. کنترل RadioButton (RadioButton)

این کنترل، دکمه‌های رادیویی را ایجاد می‌کند که می‌توان فقط یک گزینه (یکی از آنها) را انتخاب کرد. یعنی، برای ایجاد دکمه‌هایی به کار می‌رود که دارای گزینه‌های ناسازگار هستند. چنانچه بخواهید چند گروه از RadioButton داشته باشید، آنها را باید در کنترل‌های GroupBox مختلف اضافه کنید.

۹-۸-۱. کنترل GroupBox (GroupBox)

این کنترل برای ایجاد گروه‌های متعدد کنترل‌ها به کار می‌رود. اگر بخواهید چند کنترل را در یک گروه قرار دهید، باید آنها را به یک کنترل GroupBox اضافه کنید. خواص، متدها و رویدادهای این کنترل مانند کنترل‌های دیگر است.

۱۰-۸-۱. کنترل MenuStrip (MenuStrip)

این کنترل برای ایجاد منو به کار می‌رود. چنانچه این کنترل را به فرم اضافه کنید. شکل ۲-۱ ظاهر می‌شود که می‌توانید گزینه‌های منو را در بخش Type Here تایپ کنید. در مثال ۱ - ۱، چگونگی اضافه کردن منو را خواهید دید. این کنترل دارای خواص زیر است:

- **خاصیت Enabled**: تعیین می‌کند آیا گزینه منو فعال باشد یا غیرفعال.
- **خاصیت Checked**: تعیین می‌کند آیا گزینه منو می‌تواند دارای علامت تیک (✓) باشد یا خیر.
- **خاصیت ShortCut**: کلید میانبری را برای گزینه منو تعیین می‌کند.
- **خاصیت Text**: عنوان گزینه منو را تعیین می‌کند، ولی اگر در این خاصیت مقدار - را وارد کنید، گزینه منو به عنوان یک جدا کننده در نظر گرفته می‌شود.

شکل ۲-۱ اضافه کردن منو.

۱۱-۸-۱. کنترل ContextMenuStrip (ContextMenuStrip)

این کنترل همانند کنترل MenuStrip می‌باشد، با این تفاوت که برای ایجاد منوی میانبر به کار می‌رود. منو میانبر، منویی است که با کلیک راست بر روی کنترل نمایش داده می‌شود و می‌توان گزینه‌های آن را اجرا نمود. خواص، رویدادها و متدهای این کنترل مانند کنترل MenuStrip است.

۱۲-۸-۱. کنترل PictureBox (PictureBox)

این کنترل برای نمایش تصاویری از قبیل نمادهای گرافیکی، تصاویر بیت نگاشت[1]، شبه فایل، آیکن‌ها و غیره به کار می‌رود. برخی از خواص و متدهای این کنترل در زیر آمده‌اند:

خاصیت Image: تصویری را تعیین می‌کند که باید در کنترل PictureBox نمایش داده شود. به عنوان مثال، دستورات زیر را ببینید:

```
Dim img1 As Image = New Bitmap ("D:\1.gif")
PictureBox1.Image = (Image) img1
```

این دستورات، فایل 1.gif ریشه درایو D را در کنترل PictureBox1 نمایش می‌دهند.

خاصیت SizeMode: نحوه و اندازه نمایش تصویر را در PictureBox تعیین می‌کند و می‌تواند مقادیر زیر را بپذیرد:

۱. **مقدار Normal**: اندازه تصاویر را تغییر نمی‌دهد (اندازه واقعی آن را نمایش می‌دهد).

۲. **مقدار StretchImage**: تصویر را به اندازه PictureBox بزرگ یا کوچک می‌کند.

۳. **مقدار AutoSize**: کنترل PictureBox را به اندازه تصویر تغییر می‌دهد.

۴. **مقدار CenterImage**: تصویر را در وسط کنترل PictureBox نمایش می‌دهد.

متد Save: برای ذخیره تصویر موجود در کنترل PictureBox بر روی حافظه جانبی به کار می‌رود. به عنوان مثال، دستور زیر را ببینید:

```
PictureBox1.Image.Save ("test.gif")
```

تصویر موجود در PictureBox1 را در فایل test.gif ذخیره می‌کند.

۹-۱. ساختارهای کنترلی

در برنامه‌های ساده، دستورات برنامه به صورت پشت سر هم (از اولین دستور به آخرین دستور) اجرا می‌شوند. گاهی نیاز است بعضی از دستورات چندین بار اجرا شوند، تحت شرایط خاصی اجرا شده یا اجرا نگردند. برای پیاده‌سازی چنین برنامه‌هایی از ساختارهای کنترلی استفاده می‌شود. این ساختار دو نوع‌اند که عبارتند از:

۱. ساختارهای تصمیم

۲. ساختارهای تکرار

۱-۹-۱. ساختارهای تصمیم

این ساختارها برای حالتی به کار می‌روند که بخواهید تحت شرایطی مجموعه‌ای از دستورات اجرا شوند یا برخی دیگر اجرا نگردند. ساختارهای تصمیم در ویژوال بیسیک‌نت عبارت‌اند از: if و Select Case.

---

[1]- Bitmap

ساختار تصمیم if

این ساختار یک عبارت شرطی را ارزیابی می‌کند، در صورتی که نتیجه ارزیابی شرط دارای ارزش درست باشد، مجموعه‌ای از دستورات اجرا می‌شوند، وگرنه، مجموعه دیگری از دستورات اجرا خواهند شد. این ساختار به صورت زیر به کار می‌رود:

```
if  عبارت شرطی  Then
        مجموعه دستورات ۱
Else
    مجموعه دستورات ۲
End If
```

در این ساختار، ابتدا **عبارت شرطی** ارزیابی می‌گردد، اگر نتیجه ارزیابی درست باشد، **مجموعه دستورات۱** اجرا می‌شوند، وگرنه **مجموعه دستورات۲** اجرا خواهند شد.

در این ساختار به نکات زیر توجه کنید:

- در این ساختار می‌توان بخش Else را حذف کرد. در این صورت، اگر نتیجه **ارزیابی عبارت شرطی** نادرست باشد، دستورات بعد از End If اجرا خواهند شد.
- در این ساختار، عبارت شرطی می‌توان با عملگرهای منطقی از قبیل AND ، OR، XOR ، EQV و IMPشرط‌های مختلف را ترکیب کرد.

ساختار If تودرتو

گاهی ممکن است بخواهید شرط‌های متعددی را بررسی کنید. برای این منظور، می‌توانید از ساختار if تودرتو استفاده کنید. این ساختار به صورت زیر به کار می‌رود:

```
If        عبارت شرطی۱ Then
          مجموعه دستورات ۱
ElseIf     عبارت شرطی۲ Then
          مجموعه دستورات ۲
          ⋮
ElseIf   n  عبارت شرطی Then
          مجموعه دستوراتn
Else
          مجموعه دستورات1+n
End If
```

در این ساختار، اگر نتیجه **عبارت شرطی۱**، درست باشد، **مجموعه دستورات۱** اجرا می‌شوند و کنترل اجرای برنامه به بعد از End If انتقال می‌یابد. وگرنه، اگر **نتیجه عبارت شرطی۲**، درست باشد، **مجموعه**

دستورات۲ اجرا می‌شوند و دستور If خاتمه می‌یابد و این روند ادامه می‌یابد و اگر هیچ یک از عبارت‌های شرطی نتیجه درستی نداشته باشند، **مجموعه دستورات n+1**، اجرا خواهند شد.

ساختار Select Case

ساختار تصمیم تودرتو، موجب کاهش خوانایی برنامه خواهد شد. برای رفع این مشکل، می‌توان از ساختار Select Case استفاده نمود. این ساختار به صورت زیر به کار می‌رود:

```
Select Case عبارت
Case <عبارت ۱>
        دستورات ۱
Case < عبارت ۲ >
        دستورات۲
        ⋮
Case <عبارت n>
        دستورات n
Case Else
        دستورات n+1
End Select
```

در این ساختار، ابتدا عبارت جلوی Select Case، ارزیابی می‌شود. اگر نتیجه ارزیابی این عبارت، برابر با **عبارت۱** باشد، **دستورات۱** اجرا خواهند شد و ساختار Select Case خاتمه می‌یابد. اگر نتیجه ارزیابی عبارت برابر **عبارت** ۱ نباشد، با **عبارت ۲** مقایسه می‌گردد، اگر برابر این **عبارت** باشد، **دستورات۲** اجرا می‌شوند و ساختار Select Case خاتمه می‌یابد و این روند ادامه می‌یابد. اگر نتیجه **ارزیابی عبارت** برابر هیچ یک از مقادیر ۱ تا n نباشد، دستورات n+1 اجرا خواهند شد.

در ساختار Select Case باید به نکات زیر دقت کنید:

- مقادیر موجود در caseهای دستور Select Case نمی‌توانند با هم مساوی باشند.
- در جلوی Case، عملگرهای To، Is و کاما را می‌توانید با یکدیگر ترکیب کنید. دستور زیر را ببینید:

```
Case   1, 7, 9  , 12 To 22,  Is > 200
```

اگر مقدا عبارت یکی از مقادیر ۱، ۷، ۹، ۱۲ تا ۲۲ یا بزرگتر از ۲۰۰ باشد، دستور بعد از Case اجرا خواهد شد.

۱۰-۱. ساختارهای تکرار

برای انجام کارهای تکراری در برنامه از ساختارهای تکرار استفاده می‌شود. در ویژوال بیسیک‌نت حلقه‌های تکرار متعددی وجود دارند که برخی از آنها عبارتند از: For، While، Do While و For Each.

ساختار تکرار For

این ساختار برای حالتی به کار می‌رود که تعداد تکرار از قبل مشخص باشد. این ساختار به صورت زیر به کار می‌رود:

For اندیس حلقه = مقدار اولیه To مقدار نهایی Step گام حرکت
مجموعه دستورات بدنه حلقه
Next اندیس حلقه

در این ساختار، ابتدا مقدار اولیه در اندیس حلقه قرار می‌گیرد. سپس شرط حلقه تست می‌گردد، اگر شرط دارای ارزش درستی باشد، **مجموعه دستورات** بدنه حلقه اجرا می‌گردند و گام حرکت به **اندیس حلقه** اضافه یا کم می‌شود. در ادامه شرط حلقه تست می‌گردد و این روند ادامه می‌یابد. تا زمانی که شرط حلقه نقض نگردد، حلقه ادامه می‌یابد. به عنوان مثال، دستورات زیر را ببینید.

```
Dim sum As Integer = 0
For i = 1 to 100   step 1
   sum += i
   listBox1.Items.Add (i.ToString())
Next i
textBox1.Text = sum.ToString()
```

این دستورات، اعداد ۱ تا ۱۰۰ را به listBox1 اضافه کرده، مجموع آنها را در textBox1 نمایش می‌دهند.

ساختار تکرار while

این ساختار برای زمانی به کار می‌رود که تعداد تکرار از قبل مشخص نباشد و به صورت زیر استفاده می‌شود:

While شرط
مجموعه دستورات بدنه حلقه
End While

در این ساختار، ابتدا شرط ارزیابی می‌شود. اگر **شرط درست** باشد، **مجموعه دستورات بدنه حلقه** اجرا می‌شوند و شرط مجدداً ارزیابی می‌گردد و این روند تا زمانی که شرط ارزش درستی داشته باشد، ادامه می‌یابد و به محض اینکه شرط نقض (نادرست) شود، حلقه خاتمه می‌یابد. نکته‌ای که باید دقت کرد، این است که شرط باید در داخل حلقه نقض شود، وگرنه حلقه بی‌نهایت ایجاد می‌شود.

ساختار تکرار Do While

این ساختار همانند ساختار while است. با این تفاوت که **دستورات بدنه حلقه حداقل** یک بار اجرا می‌گردند. چون، شرط در انتهای حلقه ارزیابی (تست) می‌شود. این ساختار به صورت زیر به کار می‌رود:

```
Do
    مجموعه دستورات بدنه حلقه
شرط Loop While
```

دستور Exit For

این دستور برای خروج از حلقه تکرار For استفاده می‌شود و به صورت زیر به کار می‌رود:

```
Exit For
```

دستور Exit While

این دستور برای خروج از حلقه تکرار While استفاده می‌شود و به صورت زیر به کار می‌رود:

```
Exit  While
```

۱۱-۱. مدیریت صفحه کلید

کنترل‌ها علاوه بر این که به رویدادهای ماوس پاسخ می‌دهند، می‌توانند به رویدادهای صفحه کلید از قبیل KeyPress، KeyDown و KeyUp پاسخ دهند. رویداد KeyPress قبل از رویداد KeyDown و رویداد KeyDown قبل از رویداد KeyUp اتفاق می‌افتد. **رویداد KeyPress**، زمانی اتفاق می‌افتد که کاربر کلیدی (به جز کلیدهای Tab، مکان‌نما و کلیدهایی که کداسکی آنها بین ۰ تا ۳۱ است) را فشار دهد. این رویداد دو پارامتر دارد که عبارتند از:

- **Sender**: از نوع Objcet است و شی‌ای را تعیین می‌کند که این رویداد بر روی آن رخ داده است.
- **e**: از نوع KeyEventArgs دارای ساختاری می‌باشد که اطلاعات کلید فشرده شده از قبیل KeyChar (مقدار کارکتر فشرده شده) و Handled (گرداننده رویداد KeyPress) را نگهداری می‌کند.

رویداد KeyDown: وقتی کاربر کلیدی را فشار می‌دهد، این رویداد اتفاق می‌افتد (این رویداد به کلیدهای مکان‌نما، Tab و کارکترهایی که کد آنها بین ۰ تا ۳۱ باشد، نیز پاسخ می‌دهد). این رویداد همچنین مانند رویداد KeyPress دارای دو پارامتر sender و e است. ساختار پارامتر e در جدول ۱۳-۱ آمده است.

رویداد KeyUp: زمانی رخ می‌دهد که کلید فشرده شده را رها کنید. پارامترهای این رویداد مانند رویداد KeyDown است.

جدول ۱۳-۱ ساختار پارامتر e مربوط به رویدادهای KeyUp و KeyDown.

خاصیت	هدف
Alt	آیا کلید Alt فشرده شده است یا خیر؟
Handled	گرداننده رویداد KeyDown و KeyUp را تعیین می‌کند.
Ctrl	آیا کلید Ctrl فشرده شده است یا خیر؟
KeyCode	کداسکی کارکتر فشرده شده را مشخص می‌کند.
KeyData	داده مربوط به کارکتر فشرده شده را تعیین می‌کند.
KeyValue	کارکتر فشرده شده را مشخص می‌نماید.
Shift	آیا کلید Shift فشرده شده است یا خیر؟

۱۲-۱. آرایه‌ها

تاکنون متغیرهایی که در برنامه‌ها استفاده کردیم، یک مقدار را ذخیره می‌کردند. گاهی نیاز است چندین مقدار از یک نوع داده را ذخیره کنید. برای این منظور باید متغیرهای **اندیس‌دار** یا **آرایه** را تعریف نمایید. آرایه‌ها با توجه به تعداد اندیس‌هایشان به انواع مختلف تقسیم می‌شوند. آرایه با یک اندیس را **آرایه یک بعدی**، آرایه دارای دو اندیس را **آرایه دو بعدی** و آرایه که n اندیس داشته باشد، **آرایه n بعدی** نام دارد. برای استفاده از آرایه دو کار باید انجام شود:

۱. اعلان آرایه

۲. تخصیص فضا به آرایه

برای ا علان آرایه یک بعدی به صورت زیر عمل می‌شود:

Dim نام آرایه As (تعداد عناصر) نوع آرایه

اکنون دستورات زیر را در نظر بگیرید:

```
Dim  a(4) As Integer
```

این دستور ، آرایه‌ای با ٤ عنصر تعریف می‌کند (مانند شکل زیر):

0	0	0	0
a(0)	a(1)	a(2)	a(3)

در هنگام استفاده از آرایه به نکات زیر توجه داشته باشید:

۱. همان‌طور که می‌دانید، آرایه متغیری اندیس‌دار است. بنابراین، برای بازیابی و مقداردهی (دستیابی) به عناصر آن از اندیس به صورت زیر استفاده می‌شود:

نام آرایه (اندیس آرایه)

اکنون دستورات زیر را ببینید:

```
Dim  a(3) As Integer
a [0] = 10
a [1] = 12
a [2] = a [1] + a [0]
```

دستور اول، آرایه‌ای به نام a با ۳ عنصر تعریف می‌کند. دستور دوم، اولین عنصر آرایه (اندیس صفر) را برابر ۱۰ قرار می‌دهد. دستور سوم، دومین عنصر آرایه را برابر ۱۲ قرار داده و چهارمین دستور، مجموع اولین عنصر و دومین عنصر (۱۲+۱۰) را در سومین عنصر آرایه قرار می‌دهد.

۲. برای تعیین تعداد عناصر آرایه از خاصیت Length استفاده می‌شود. دستورات زیر را ببینید:

```
Dim  a(5) As Integer
label1.Text = a.Length
```

دستور اول، آرایه‌ای با ۵ عنصر تعریف می‌کند. دستور دوم، ۵ را در label1 نمایش می‌دهد (تعداد عناصر آرایه a را با خاصیت Length تعیین می‌کند).