**أساسيّات اللغة**

لا غنى ـ حتّى لمبرمجي VB6 المحترفينَ ـ عن قراءة هذا الجزء من الكتاب، فهناك تغييراتٌ واسعةُ النطاقِ وإضافاتٌ لا حصرَ لها طرأت على VB6.

1. [**التعرّف على اللغة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\01-%20التعرف%20على%20اللغة.htm)

- التعرّف على بيئة التطوير المتكاملة، وأهمّ نوافذها وقوائمها الرئيسيّة.

- التعرّف على بعض الأدوات والخصائص والوسائل والأحداث المشتركة بينها.

1. [**المشاريع:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\02-%20المشاريع.htm)

- إنشاء مشروع "محاسب القروض" ومشروع "الآلة الحاسبة".

- التعرّف على كيفيّة تصميم واجهة الاستخدام.

- التعرّف على كيفيّة التحقّق من صحّة البيانات، وتتبّع الأخطاء وإصلاحها وكتابة الكود الذي يقي من أخطارها.

- إنشاء برنامج إعداد Setup لتوزيع المشاريع على أجهزة أخرى.

- التعرّف على تطبيقات الإنترنت.

1. [**المتغيّرات Variables:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\03-%20المتغيرات.htm)

- التعرّف على أنواع المتغيّرات وكيفيّة تعريفها واستخدامها.

- التعرّف على المصفوفات Arrays.

- التعرّف على كيفيّة بناء الأنواع الخاصّة Use-Defined Types، مثل المرقّمات Enums والسجلات Structures.

1. [**مسار البرنامج وأجزاؤه:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\04-%20مسار%20البرنامج%20وأجزاؤه.htm)

- التعرّف على الجمل الشرطيّة والجمل التكراريّة Loops.

- التعرّف على كيفيّة بناء الإجراءات الفرعيّة والدوال Functions.

1. [**النماذج Forms:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\05-%20Forms%20النماذج.htm)

- التعرّف على خصائص النموذج ووسائله.

- التعرّف على كيفيّة التعامل مع النموذج وبرمجته.

- التعرّف على كيفيّة إنشاء القوائم الرئيسيّة Menus والقوائم الموضعيّة Context Menus.

- التعرّف على واجهة الوثائق المتعدّدة MDI والنماذج المترابطة.

1. [**الأدوات الأساسيّة Basic Controls:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20أساسيّات%20اللغة\06-%20Basic%20Controls%20الأدوات%20الأساسيّة.htm)

- التعرّف على خصائص ووسائل وأحداث الأدوات التالية:

- مربّع النصّ TextBox.

- القوائم Lists.

- الفاصل Splitter.

- أداة تلميحات الشاشة ToolTip.

- المنزلق ScrollBar.

- شريط القيمTrackBar .

- التعرّف على السحب والإسقاط Drag And Drop.

**الفصل الأول**

**التعرّف على اللغة**

* [**مقدمة في عجالة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20مقدمة%20في%20عجالة.htm)

كما فهم اللمّاحون منكم، يمثّل هذا المقطع مقدّمةً عاجلةً عن اللغة، وعن هذا الفصل!

* [**بيئة التطوير المتكاملة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة.htm)

واضحٌ أنّ هذه العناوين لا تناسب إلا العباقرة، فمَن غيرهم سيدرك أنّ هذا المقطع مخصّصٌ للتعرّف على بيئة التطوير المتكاملة؟!!

سنتعرّف على صفحة البداية Start Page، ونحاول أن نتحسّس خطانا الأولى في واجهة VS.Net، الجديدة علينا.

* [**مشروعك الأول في VB.NET:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول.htm)

هل تدرك بهجة الأمّ حينما يقول ابنها كلمة "كخّ" لأوّل مرّة في حياته؟!.. هنا ستدرك ما تحمله هذه الكلمة من إبداع خلاق، عندما ننشئ تطبيقا يكتب فيه المستخدم اسم لغة البرمجة، فنشفعه بتعليق من عندنا على غرار: "اختيار ممتاز"!!.. وكما سيطوّر الطفل قدراته على النطق، فيرتقي إلى كلمة "كخّة"، سنطوّر نحن تطبيقنا لتلافي بعض الأخطاء المحتملة!

أعتقد أنّك ستستمتع بهذا التطبيق، لأنّ البرمجة ما زالت بالنسبة لك قارةً خصبةً مجهولة، وكلّ سطرٍ تكتبه من الكود يعدّ اكتشافا جديدا في مجاهلها (أرجو ألا تبيد كلّ هنود هذه القارّة، كما فعل أصدقاؤنا إيّاهم، دعاة الحضارة وحقوق الإنسان!!).

* [**مكونات بيئة التطوير:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\4-%20مكونات%20بيئة%20التطوير.htm)

هذه هي الخطوط العامّة، نضعها على خارطة القارّة الجديدة لتمثّل أبعادها وحدودها.. سنتعرّف ها هنا على أهمّ قوائم واجهة بيئة التطوير، وأهمّ نوافذها، وأهمّ خياراتها.

* [**تعدد الأشكال Polymorphism:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\5-%20تعدد%20الأشكال%20Polymorphism.htm)

سيبدو لك الآن أنّك تركب صاروخ فضاء لاكتشاف الفضاء الخارجيّ!.. هنا نتعرّف على أهمّ الخصائص Properties والوسائل Methods والأحداث Events المشتركة بين أدوات VB.

* [**أنواعُ المشاريع:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\01-%20التعرّف%20على%20اللغة\6-%20أنواعُ%20المشاريع.htm)

واضحٌ أنّ هذا الفصل ملئ بالعناوين المثيرة للخيال!.. هذا المقطع بالفعل يتحدّث عن أنواع المشاريع التي يمكنك إنشاؤها بـ VB.Net.

**مقدمة في عجالة:**

أهلا بك في VB.NET.

لو لم تكن تعرف أنك محظوظ فاعرف، فأنت الآن تتعلّم البرمجةَ في الوقت الذي صارت فيه متعةً، لما اكتسبته من سهولة ويسر وقوّة.

فبخلاف أنّك تكتب بلغة VB، وهي تمتاز بسهولة أوامرها، وتركيبها الذي يجعلها قريبة من اللغة الإنجليزية العادية، فإنّك الآن تستخدم VS.NET، التي صمّمت لتحاول أن توفر لك ثروةً من الأدوات والدوال الجاهزة، التي تريحك من عناء البدء من الصفر، والتي تعلو بك كذلك عن تركيب الجهاز بل ونظام التشغيل، بحيث يستطيع برنامجك أن يعمل على أي نظامٍ، دونَ أن تخشى من مشاكل عدم التوافق Incompatibility بينهما.

وحتّى تكونَ مطمئنًّا، فإنَّ هذا الكتابَ لا يتطلّبُ سابقَ معرفةٍ بلغة VB6.. بل لا يتطلّبُ حتى سابق معرفة بالبرمجة!

إنَّ VS.NET هي بيئةُ تطويرٍ متكاملةٌ Integrated Development Environment (IDE)، لبناءِ واختبارِ وتصحيحِ البرامجِ المتنوّعة: تطبيقات ويندوز Windows Applications، تطبيقات الإنترنت Web Applications، الخلايا Classes، وأدوات التحكم الخاصّة Custom Controls، وهي تمنحُك العديدَ من الأدواتِ المرئيّةِ Visual Tools لإنشاء واجهة التطبيق Application Interface بشكلٍ شبه آليّ، ولتسريع وتسهيل أداءِ العديدِ من المهامِّ في التصميمِ والبرمجةِ.

لهذا فقد خُصّص هذا الفصلُ ليجعلكَ تألفُ هذه البيئة، وتعرفُ كيفَ تُساعدُ أدواتُها على سرعةِ وسهولةِ تصميمِ واجهةِ البرنامج، وكتابةِ الكود.

ونحنُ هنا بصددِ أن نتعرّفَ على المكوناتِ الرئيسيّةِ لبيئةِ التطويرِ المتكاملةIDE ، خاصّةً تلك التي سنحتاجُ إليها لبناءِ تطبيق ويندوز بسيط.

أعتقدُ أنّك قد أنزلت هذه اللغةَ على جهازكَ بالفعل، وربما تكونُ قد استكشفتَ واجهتَها الجديدة.

وإذا لم تكن قد فعلت فافعل، لأن قراءة هذا الكتاب بدون تطبيقه عمليا خطوة خطوة، لن تأتي بالفائدة المرجوّة (يمكنك الاستعانة بالخطوات الموضّحة في مقدّمة الكتاب لإعداد هذه اللغة على جهازك).

**ملحوظة هامّة جدا جدّا:**

في أيّ مكان من بيئة التطوير: في صندوق الأدوات، في نافذة الخصائص، في مصمّم النماذج، في محرّر الكود... إلخ، حدّد العنصر الذي تريد الاستعلامَ عنه بضغطه بالفأرة مرّة، أو حدّد حروف الكلمة التي تريد الاستعلام عنها بنقرها مرتين بالفأرة، ثم **اضغط زر F1**، لتظهر لك نافذة استعلامات VS.NET، حيث ستجد المعلومات التي تحتاج إليها.

**بيئة التطوير المتكاملة:**

* [**ما هي:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\0-%20ما%20هي.htm)
* [**صفحةُ البداية Start Page:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\1-%20صفحةُ%20البداية.htm)
* [**بدء مشروعٍ جديد:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\2-%20بدء%20مشروعٍ%20جديد.htm)
* [**في بيئة التطوير IDE:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\3-%20في%20بيئة%20التطوير.htm)
* [**استخدام مصمم نماذج ويندوز Windows Form Designer:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\4-%20استخدام%20مصمم%20نماذج%20ويندوز.htm)
* [**هذه أداتي الأولى:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\5-%20هذه%20أداتي%20الأولى.htm)
* [**تجربة البرنامج:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20بيئة%20التطوير%20المتكاملة\6-%20تجربة%20البرنامج.htm)

**ما هي:**

صُمّمت VS.NET لاستضافةِ أيِّ لغةٍ من لغاتِ البرمجة، وهناكَ الآنَ بالفعلِ عديدٌ من الشركاتِ التي تعملُ على تطويرِ لغاتٍ مختلفةٍ لتتوافقَ مع VS.NET، وقريبًا ستجدُ أنّ بعضَنا يبرمجُ تطبيقات ويندوز في VS.NET بلغة كوبول أو لغة فورتران!

إنَّ الواجهةَ المرئيّةَ للبرنامجِ ليست مرتبطةً بلغةِ برمجةٍ بعينِها، ونفسَ الأدواتِ التي ستستخدمُها لتطويرِ واجهةِ تطبيقك، ستجدُ أنَّ كلَّ المبرمجينَ يستخدمونَها مهما اختلفتِ لغةُ البرمجةِ التي يستخدمونَها لكتابةِ الأوامر!

ينطبقُ نفسُ الأمرِ كذلك على الأدواتِ التي ستستخدمُها للاتصالِ بقواعدِ البيانات والتعامل معها، فهي غير معتمدةٍ على اللغةِ التي تستخدمُها.

إنّ جعلَ بيئةِ التطويرِ مستقلّةً عن لغاتِ البرمجة، يمنحُ المبرمجينَ سهولةً في تعلّم أيّ لغة، حيث لن يضطرّوا لتعلّم البيئة الخاصة بكل لغة جديدة يقدمون على تعلّمها، كما أنّ ذلك يجعلُ استخدام كائناتٍ Objects كتبت بلغاتٍ مختلفةٍ أمرًا أيسر وأكثرَ توافقًا.. ليس هذا فحسب، بل يتيح كذلك إنشاء تطبيقات تستخدم أجزاءً مكتوبة بلغات برمجة مختلفة، مع إمكانيّة فتح كلّ هذه الأجزاء معا لتصميمها في وقت واحد!

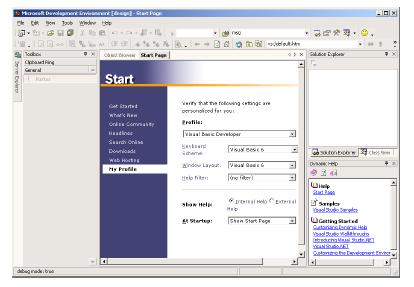
هذا، وتحتوي بيئةُ التطويرِ المتكاملةُ IDE على العديدِ من الأدواتِ المرئيّةِ، كمصمّم القوائمِ الرئيسية Menu Designer، الذي يساعدكُ على إنشاءِ القوائمِ الرئيسية Menus، وتحديدِ أسمائها وخواصّها، وما يندرج تحتها من أوامر Commands و قوائم فرعية Submenus، كلّ ذلك بطريقةٍ مرئيّة (أي باستخدام الفأرةِ Mouse ولوحةِ المفاتيحِ Keyboard ونافذة الخصائص Properties Window، بدلا من كتابةِ كود لذلك).

كما تحتوي بيئةُ التطوير كذلك على الأدواتِ التي تُمكّنك من تصميمِ Design وتنفيذ Execute وتصحيح أخطاء Debug برامجك.

إنّ الأمرَ يبدو كما لو أنّها صارت نظامَ تشغيلك الثاني!

**صفحةُ البداية Start Page:**

عندما تفتح VS.NET لأوّل مرّة، ستطالعك صفحة البداية.



اضغط الاختيار "My Profile"، حتّى يمكنَك أن تحدّد خياراتك الأوّليّة باختيار لغة البرمجةِ التي ستعمل بها.. اختر "مبرمج فيجيوال بيزيك" Visual Basic Developer في مربع "الملمح" Profile، وستجدُ أنَّ المربعينِ الآخرينِ قد تم ملؤهما آليًّا.. يمكنُكَ أن تتركَ القيمَ الافتراضيّةَ في باقي الخاناتِ.

كما يمكنُكَ أن تحدّدَ ما سيحدثُ عن فتحِ VS.NET، عن طريقِ القائمةِ المنسدلة "في بدء التشغيل" At Startup أسفلَ الصفحة.. لديك الخياراتُ التالية:

**- اعرض صفحة البداية Show Start Page:**

حيثُ ستظهرُ صفحةُ البدايةِ في كلِّ مرّةٍ تفتحُ فيها VS.NET.

**- افتح آخر مشروع Load Last Loaded Solution:**

حينما تجدُ نفسَكَ غارقًا لشهورٍ وأسابيعَ في تطويرِ برنامجٍ ضخم، اخترْ هذا الاختيارَ لفتحِ مشروعِكَ تلقائيًّا كلما فتحت VS.NET.

**- اعرض مربع حوار "فتح مشروع" Show Open Project Dialog Box:**

في كلِّ مرّةٍ تفتحُ فيها VS.NET، سيظهرُ لك مربع حوار "فتح مشروع"، حيثُ يمكنُكَ أن تختارَ المشروعَ الذي تريدُ فتحَه.

**- اعرض مربع حوار "مشروع جديد" :Show New Project Dialog Box**

في كلِّ مرّةٍ تفتحُ فيها VS.NET، سيظهرُ لك مربع حوار "مشروع جديد"، حيثُ يمكنُكَ أن تختارَ نوعَ المشروعِ وتحدّد اسمَه.. ننصحُك بتجنّبِ هذا الاختيار.

**- اعرض بيئةَ التطوير فارغةً Show Empty Environment:**

ستبدأ VS.NET ببيئةٍ فارغة، وعليك أن تفتحَ أو تنشئَ المشاريعَ التي تريدُ التعاملَ معها.

إنَّ الخيارَ الأكثر شيوعًا هو الخيارُ الأوّل، حيثُ إنَّ صفحةَ البدايةِ تعرضُ آخرَ أربعة مشاريعَ تمَّ فتحُها، بالإضافةِ لزرّي "مشروع جديد" و"فتح مشروع".

ولكي ترى صفحةَ البدايةِ، فإنَّ عليكَ أن تحدّد خيار Get Started"".

أمّا باقي الخيارات، فهي تُشيرُ لمواقع VS.NET على الإنترنت، حيثُ توجدُ أحدثُ المعلوماتِ عن المنتجات، وكذلك أحدثُ المقالاتِ والوثائق، وحزمُ الخدمات.

وعليك ما بينَ الحينِ والآخرِ، أن تحاولَ تحديثَ نسختِك من VS.NET.

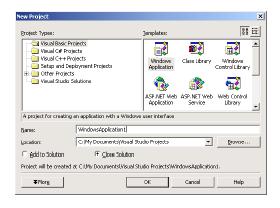
أمّا خيار Web Hosting""، فإنّه يقودُ لصفحةٍ تحتوي على معلوماتٍ عن مواقع الخدمات التي تدعّم ASP.NET..

**بدء مشروعٍ جديد:**

نحنُ الآنَ في سبيلنا لإنشاءِ مشروعنا الأول، لتوضيح العناصر المختلفة لبيئة التطوير.. وسنكتفي في البداية بإنشاءِ نموذجٍ Form فقط.

والنموذج هو النافذة التي يعرضها برنامجك، حيث يراه المستخدمون على شاشة الجهاز عند تشغيل البرنامج.

اضغط القائمة الرئيسية "ملف" File واختر "مشروع جديد" New Project.. في مربع حوار "مشروع جديد" سترى قائمةً من أنواع المشاريع التي يمكن إنشاؤها بـ VB.NET.



اختر النوع "تطبيق ويندوز"Windows Application.. سيقترح VB.NET اسمًا افتراضيًا للمشروع WindowsApplication1.. غيّره إلى MyTestApplication.

**ملحوظة:**

رغم أن تسمية المشروع والنماذج والمتغيرات بحروف عربية مسموح به، إلا إننا ننصحك بتسميتها بحروفٍ أجنبية، وذلك حتّى لا تُضطر أثناء كتابة الكود إلى كثرة التغيير بين اللغةِ العربية والإنجليزية، كما أنّ بعض أنظمة التشغيل تسبب مشاكل مع الحروف العربية.. ويمكنك في المقابل أن تسّمى عناصرك بمنطوقٍ عربيٍّ لكن بحروف أجنبية، مثل Pernamagy، بدلا من برنامجي.

تحت خانة اسم المشروع Project name، توجد خانة الموقع Location، حيثُ يمكنك هنا أن تحدّد مسار واسم المجلّد الذي سيُنشَأ فيه البرنامج.

**ملاحظتان:**

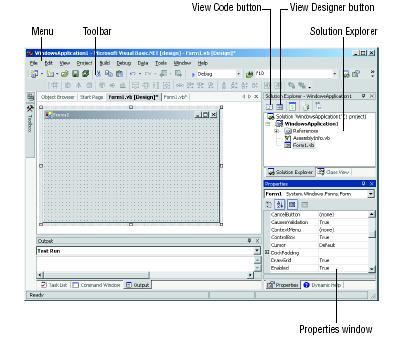
1- على خلافِ VB6، يحفظ VB.NET كل مشروع في مجلد خاص به له نفس اسم المشروع، يتمّ إنشاؤه داخل المجلد الذي حدّدته أنت في خانة الموقع.

2- ستجدُ أن VB.NET يقترح عليك مجلدا افتراضيا في خانة الموقع.. ننصحكُ أن تغيّرَه، لأنّه في الغالب سيكونُ في القسمِ من القرص الصلب Hard Disk الذي أنزلتَ عليه نظام التشغيل Operating System، مّما يهدّد بضياعه وكلّ ما يحتويه من برامج في حالة حدوث خللٍ في نظامِ التشغيلِ يحتاجُ لإعادةِ تنصيبه على الجهاز.. لهذا تخيّرِ الموقعَ على أيِّ جزءٍ آخرَ من القرصِ الصُلبِ بعيدٍ عن نظام التشغيل.

لإغلاقِ مربع الحوار وتنفيذ ما به، اضغط زر موافق OK.

**في بيئة التطوير IDE:**

إنّ ما تراه الآن هو بيئةُ التطبيق وقد عرضت مصمم النماذج Form Designer للمشروع الجديد.



النافذة الرئيسية هي مصمم النماذج، وسطحه الرماديُّ هو نافذة تطبيقك الجديد في طور التصميم Design mode.

ويمكنك بواسطة مصمم النماذج، أن تصمم الواجهة المرئية للتطبيق، حيثُ يمكنكُ أن تضعَ العديد من مكونات واجهة ويندوز Windows Interface Components على النموذج، كما يمكنك أن تبرمج التطبيق بكتابة أوامره.

وبطبيعة حالها، فإن بيئة التطوير في وضعها الافتراضي Default stateمزدحمة نوعًا، لهذا دعنا نُخفي قليلا من أشرطة الأدوات Toolbars التي لن نستخدمها في الفصول القليلة الأولى.. وبالطبعِ تستطيعُ أن تعيد إظهارِ أي شريط أدواتٍ في أي وقت تشاء.

افتح القائمة الرئيسية "عرض" View، واضغط الأمر "أشرطة الأدوات" Toolbars.. ستظهر لك قائمة فرعية، تحتوي على أسماء أشرطة الأدوات، وبجوار بعضها علامة الاختيار، مما يعني أنّها معروضة حاليا.. اضغط على أيِّ اسم شريطٍ منها لتغيير حالته، فلو كان مختارا فستزول عنه علامة الاختيار ويختفي من الشاشة، ولو كان غير مختارٍ، فسيكتسب علامة الاختيار ويظهر على الشاشة.

أخْفِ جميع أشرطة الأدواتِ ما عدا شريطَيْ "تخطيط" Layout و"قياسي" Standard.

لعلك لاحظتَ أنّ الأمر الأخير في قائمة أشرطة الأدوات هو "تخصيص" Customize، وهو يعرضُ مربع حوارٍ Dialog box، يمكنك فيه أن تحدد ما تريد أن تراه من أشرطة الأدوات، وكذلك من الأزرار الموجودة على كل شريط منها.. حاول تجربته.. لا شيء ستخسره!

**استخدام مصمم نماذج ويندوز Windows Form Designer:**

لتصميم النموذج، عليك أن تضع عليه كل الأدوات Controls التي تريد أن تعرضها لمستخدم برنامجك في وقت التشغيل Runtime.

وأدوات التحكم Controls ـ وسنختصرها بكلمة "الأدوات" ـ هي مكونات واجهة ويندوز Windows Interface، كالأزرار Buttons، وأزرار التحويل Radio Buttons، والقوائم Lists.. إلخ.

افتح صندوق الأدوات Toolbox، بتحريك مؤشر الفأرة فوقَ شريط صندوق الأدوات Toolbox tab في أقصى يسار الشاشة.. سيبرز إليك صندوقُ الأدوات.



يحتوي صندوق الأدوات هذا على أيقونةٍ لكلِّ أداةٍ Control يمكنك استخدامها على نموذجك، حيثُ تمّ ترتيبُ وتبويب الأدوات في شرائط Tabs، على حسب أنواع المشاريع التي تستخدم فيها.

فمثلا: في الجزء الأول من هذا الكتاب، ننشئ تطبيقات ويندوز بسيطة، لهذا سنستخدم الأدوات الموجودة على شريط "نماذج ويندوز" Windows Forms.

لو أنك ضغطت شريط "نماذج الإنترنت"، فستجد أيقوناتها كلها غير فعالة Disabled.. بمعنى أنّ شكلها قد تغيّر، ولم تعد تستجيبُ لحركة الفأرةِ فوقَها، كما لم يعد من الممكن ضغطها أو استخدامها.

ولكن عندما تطوّر أحد تطبيقات الإنترنت، فستجدُ أنّ الأيقونات على شريط "نماذج ويندوز" أصبحت غير فعالة Disabled، بينما سيمكنك أن تضع على نموذج الإنترنت Web Form، أيًا من أدوات الإنترنت الموجودة على شريط "نماذج الإنترنت" Web Forms.

نعودُ لموضوعنا: ماذا تفعلُ لوضع الأدوات على نموذجك؟

ما عليك إلا أن تنقر مرتين بالفأرة Double-click على الأيقونة التي تمثّل الأداة المطلوبة، وفي الحال ستوضع على النموذج نسخةٌ جديدة من هذه الأداة، ذات موضعٍ وأبعاد افتراضيّة.

يمكنُك بعد ذلك أن تغيّر موضع الأداةِ بسحبها بالفأرة.. كما يمكنُك أن تغيّر أبعادها بأن تتحرّكَ بالفأرةِ فوقَ جوانبِها، وتتوقّف فوقَ المربّعات الصغيرةِ على محيطها.. وعندما يتحوّل مؤشر الفأرة إلى شكل سهم ذي رأسين، اضغط زرّ الفأرةَ الأيسر واسحبها في أيّ اتجاه أثناء ضغطك.. بهذا تستطيعُ أن تمدّد أو تُقلّصَ من طولِ الأداةِ في هذا الاتجاه.

**ملحوظة:**

يمكن استخدام نفس الطريقة لتغيير حجم النموذج نفسه.

يمكنك كذلك وضع نسخة جديدة New Instanceمن الأداة على النموذج، عن طريقِ ضغط أيقونتها مرة واحدة، والانتقال إلى النموذج، وضغط زر الفأرةِ الأيسر، ثم التحرك بها فوقَ النموذجِ، مع استمرار الضغط.. ستجد أنّ مستطيلا يمثّل الإطار الخارجيَّ للأداةِ يرتسمُ على النموذجِ وتتغيّر أبعاده مع حركة الفأرة.. عندما ترى أبعاد المستطيل ملائمة لما تريده، اترك زر الفأرة.. في الحال سترتسم الأداة داخلَ الإطارِ الذي حدّدته.. يمكنكُ بالطبعِ تغيير أبعادها وموضعها من جديد في أية لحظةٍ كما ذكرنا سابقا.

ولحذف أي أداة، حددها (بنقرها بالفأرةِ مرةً واحدة)، ثم اضغط "حذف" Delete من لوحة المفاتيح.

هذا، وتُعرض خصائص كل أداةٍ في نافذة الخصائص Properties window، التي توجد في أقصى يمين الشاشة، وبواسطتها يمكنك عرض وتغيير خواص الأداة المحددة حاليا على النموذج.

وإذا لم تكن نافذة الخصائص ظاهرة، فاضغط القائمة الرئيسية "عرض" View ثم اختر منها "نافذة الخصائص" Properties Window.. أو اضغط مباشرة F4 من لوحة المفاتيح.

**ملاحظة:**

يمكنك تنفيذ أي أمر في بيئة الويندوز بأكثر من طريقة:

- فإما أن تختاره من القوائم الرئيسية Menus.

- وإما أن تضغط زر الفأرة الأيمن لتظهر لك القائمة الموضعية Context Menu، حيث ستجدُ أهم الأوامر المرتبطة بالعنصر الحالي.

- وإما أن تضغط زر الأمر على أحد شرائط الأدوات Tool Bars في أعلى النافذةِ، أسفل القوائم الرئيسية مباشرة.. ولكي تعرف وظيفة كلّ زرّ، حلّق فوقه بالفأرة لحظة، وسيظهر تلميح على الشاشة TooTip يشرح لك وظيفة هذا الزرّ.. هذا إذا لم تكن أيقونة الزرّ تشرح وظيفته بوضوح.

- أو تضغط اختصار الأمر من لوحة المفاتيح، والذي يكونُ في الغالب أحد أزرار الوظائف F1, F2, .... F12، أو أي تركيب من الأحرف مع أزرار التحكم Ctrl أو Shift أو Alt.. وستجدُ مثل هذه الاختصارات مكتوبة بجوار الأمر في القوائم الرئيسية.

وإذا لم تكن هناك أداة محددة أو نموذج محددًا، فستعرض نافذة الخصائص خصائص العنصر الحالي في متصفح المشاريع Solution Explorer.

**هذه أداتي الأولى:**

والآن أضف مربع نص TextBox للنموذج.. اضغط عليه بزر الفأرة الأيمن، ومن القائمة الموضعية اختر "خصائص" Properties.

في نافذة الخصائص، يمكنك أن ترى الخصائص التي تحدد كيف تظهر الأداة على النموذج، وفي بعض الأحيان كيف تعمل.. تعال نجرّب بعض الخصائص:

**الخاصيّة "نص" Text:**

ابحث عن خاصية "نص" Text، وضع لها القيمة: "هذا هو مربع النص الخاص بي"، وذلك بكتابة هذا النص ـ بدون علامتي التنصيص ـ في الخانة الموجودة على يمين الخاصية، بدلا من النص الافتراضي "TextBox1".

إن خاصية Textهي المسئولة عما يُعرض في مربع النص من كلمات، وأي نص يُكتب بها تجده مكتوبا داخل مربع النص.

**ملحوظتان:**

1- عليّ أن أكتب الخصائص باللغة الإنجليزية، لأنها هكذا ستستخدم في كتابة الكود، وإن كنت حريصا على ترجمة معانيها للعربية، ليسهل فهمها على من لا يجيدون الإنجليزية.

2- عندما تحدد اسم أي خاصية، يظهر تعريف مختصر لها في أسفلِ نافذة الخصائص، ليساعدك على فهم وظيفتها.. كما يمكنك أن تضغط F1 لتظهر لك نافذة المساعدة Help Window، التي تشرح لك كل التفاصيل المتعلقة بهذه الخاصية، والأخطاء التي يحتمل أن يسببها وضع بعض القيم الخاطئة بها، وأحيانًا أمثلةً على استخدامها.. حاول أن تعتاد قراءة هذه المعلومات، ولا تجزع من كونها بالإنجليزية، فهي في أبسط صورها، ومعظمها مصطلحات برمجية سريعا ما ستألفها.

**خاصية "لون الخلفية" BackColor:**

ابحث عن خاصية "لون الخلفية" BackColor وحددها بالفأرة.. سيظهر لك في الخانة اليمنى زر الانتقال Ellipsis Button.. اضغط عليه.. سيظهر لك مربع حوارٍ Dialog box ذو ثلاثة أشرطة: مخصّص Custom، إنترنت Web، ونظام System.



في هذا الحوار تستطيع اختيار اللون ـ الذي سيظهر في خلفية مربع الحوار ـ من أيٍّ من الأشرطة الثلاثة.. اختر اللون الأصفر، ولاحظ التغيير الذي سيطرأ على مربع النص.

**خاصية "الخط" Font:**

ثم ابحث عن خاصية "الخط" Font.. ستجد بجوارها علامة "+".. يمكنك أن تضغطها لتظهر لك الوحدات التي تشكل خصائص الخط، حيث يمكنك أن تغير كلا منها على حدة.. أو بإمكانك أن تضغط زر الانتقال ellipsis Buttonلتعرض مربع حوار الخط Font dialog box.. في هذا الحوار يمكنك اختيار الخط وتحديد خواصه.. اختر الخط Verdana، 14 نقطة Points، سميك Bold.

**خاصية "متعدد الأسطر" MultiLine:**

إن مربع النصّ في الوضع التلقائيّ لا يتقبّل كتابةَ أكثر من سطر واحدٍ فيه.. لتغيير ذلك، حدد خاصية "متعدد الأسطر" MultiLine.. ستجد أن قيمتها الحالية هي "خطأ" False.. اضغط زر إسدال القائمة، وغيرها إلى "صواب" True.. كما يمكنك تغيير قيمة هذه الخاصية مباشرة بالنقر المزدوج Double-click على القيمة الحالية.

**خاصية "التفاف الأسطر" WordWrap:**

والآن حاول أن تكتبَ سطرًا أطول من اللازم في مربع النص.



ستلاحظ أنّ ما زاد منه عن عرض مربع النص قد التف إلى أسطرٍ إضافية، فإذا ما أزدت عرض مربع النص، فستجد أنّ السطور تتغير لتلائم العرض الجديد.. إن المسئول عن حدوث هذا هو خاصية "التفاف الأسطر" WordWrap، والتي لو حددتها في نافذة الخصائص لوجدت قيمتها True.. اجعل قيمتها False وحاول اكتشاف التغير الذي سيطرأ على محتويات مربع النص.

**خاصية "المنزلقات" ScrollBars:**

عادةً ما تحتوي مربعات النص متعددةُ الأسطر على منزلقٍ رأسيّ، حتّى يُمكّن مستخدمه من سرعةِ الوصولِ للمقطعِ الذي يهمّه في النص.

حدد خاصية "المنزلقات" ScrollBars في نافذة الخصائص، وأسدل قائمة الخيارات المتاحة.. ستجد هذه الخيارات:

|  |  |
| --- | --- |
| None | لا يعرض مربع النص أيّ منزلقات. |
| Vertical | يعرض مربع النص منزلقًا رأسيًا فقط. |
| Horizontal | يعرض مربع النص منزلقًا أفقيا فقط. |
| Both | يعرض مربع النص كلا المنزلقين الرأسيّ والأفقيّ. |

اختر Vertical، واكتب جملةً طويلةً جدًّا في خاصية Text، ور كيف يستوعبها مربع النص.

**تجربة البرنامج:**

بما أننا في وقت التصميم Design time، فلن يمكنك أن تحرك المنزلق، ولو حاولت هذا فستفاجأ بأن مربع النص نفسه يتحرك معك!.. تحتاجُ إذن لاختبار سلوك أداتنا في وقت التشغيل Runtime.. لكي تفعلَ هذا اضغط F5.. سيترجم البرنامجCompiled للغة ميكروسوفت الوسيطة Microsoft Intermediate Language (MSIL)، وبعد لحظاتٍ قليلةٍ سيظهر على الشاشة نموذج يحتوي على مربع نص.. هذا ما سيراه مستخدمو برنامجك.

أدخل بعض النصوص في مربع النص.. حدد النصوص بضغط زر الفأرة الأيمن والتحرك يمينا أو يسارا أو أعلى أو أسفل قبل ترك الزر.. ثم اضغط Ctrl+C من لوحة المفاتيح لنسخ النص الذي حددته.. إن بإمكانك الآن أن تضغط Ctrl+V، لتلصق هذا النص المنسوخ في أي موضعٍ من مربع الحوار الحالي، أو في أي أداة أخرى تقبل الكتابة، موجودة على النموذج الحالي أو أي نموذج آخر، في نفس البرنامج أو في برامج أخرى.. والعكس أيضًا صحيح، فالنصوص المنسوخة من أي تطبيق آخر، يمكن لصقها في مربع النص الحالي.. والفكرة بسيطة، حيث إن الويندوز يخزن النص المنسوخ في محزن القصاصات Clipboard، وهو مخزن عام يستطيع أي برنامج أن يكتب فيه ويقرأ منه، لذلك يُعد من أبسط الوسائل لنقل النصوص والصور بين التطبيقات المختلفة.

بعد أن تفرغَ من تجريب ذلك، اضغط القائمة الرئيسية "تصحيح" Debug، واضغط "إيقاف التصحيح"Stop Debugging .. هذا الفعل سينهي تنفيذ برنامجك، لتعود لبيئة التطوير من جديد.

**خاصية "حالة الأحرف" CharacterCasing:**

واحدة من الخصائص التي تحدد كيف يعمل مربع النص ـ ناهيك عن كيف يبدو، هي خاصية "حالة الأحرف" CharacterCasing.. وهي تأخذ القيم التالية:

|  |  |
| --- | --- |
| طبيعية Normal | تظهر الحروف المكتوبة في مربع النص كما كتبت، وهذه هي القيمة الافتراضية لهذه الخاصية. |
| كبيرة Upper | تظهر الحروف المكتوبة في مربع النص في الحالة العليا Capital. |
| صغيرة Lower | تظهر الحروف المكتوبة في مربع النص صغيرة Small. |

واضح أن هذه الخاصية لن تؤثر على الحروف العربية.

إن تصميم تطبيق جديد يبدأ بتصميم النموذج، الذي يحدد كيف يعمل التطبيق.. لهذا تحدد الأدوات الموضوعة على النموذج كيف يتفاعل التطبيق مع المستخدم.

وما النموذج نفسه إلا قالبا ابتدائيا Prototype، وبمجرد وضع الأدوات عليه وضبط خصائصها، تكون قد بنيت جزءًا كبيرًا من وظيفة برنامجك.

**مشروعك الأول في VB.NET**

* [**فكرة المشروع:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\1-%20فكرة%20المشروع.htm)
* [**نافذة محرر الكود Code Editor Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\2-%20نافذة%20محرر%20الكود.htm)
* [**"مجيبات الأحداث"Event Handlers :**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\3-%20مجيبات%20الأحداث.htm)
* [**أوّل ما تكتبه من البرمجة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\4-%20أوّل%20ما%20تكتبه%20من%20البرمجة.htm)
* [**تحسين التطبيق ليصبح أفضل أداءً:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\5-%20تحسين%20التطبيق%20ليصبح%20أفضلَ%20أداءً.htm)
* [**جعل تطبيقنا أكثر حميميةً للمستخدم More User-Friendly:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\3-%20مشروعك%20الأول\6-%20جعل%20تطبيقنا%20أكثر%20حميميةً%20للمستخدم.htm)

**فكرة المشروع:**

ها نحن أولاء بصدد إنشاء تطبيق بسيطٍ لإيضاح كيفية تصميم واجهة البرنامج وكتابة أوامره.

وتطبيقنا سيسمح للمستخدم بكتابة أسماء لغات البرمجة المفضلة لديه، ثم سنقيّم اختياره.

**ملحوظتان:**

1- يحمل هذا البرنامج اسم WindowsApplication1، ويمكنك أن تجده مع الكتاب في مجلد البرامج، داخل المجلد الذي يحمل رقم هذا الفصل.. يمكنك أن تنسخ هذا البرنامج لأي موضع آخر (حتّى لا تؤدّي التعديلات التي يمكن أن تجرّبها عليه لإفساده)، وتأكد من تغيير خاصية "للقراءة فقط" read-only للمجلد وللملفات داخله، حتى يمكنك أن تجرى في البرنامج التعديلات التي تريد.. ولعمل ذلك اتبع ما يلي:

- حدد المجلد واضغط زر الفأرة الأيمن.. ستظهر لك القائمة الموضعية.. اختر منها "خواص" Properties.. سيظهر لك مربع حوار الخواص.. أزل العلامة من مربع الخيار CheckBox "للقراءة فقط" Read-Only، واضغط "موافق" OK.

- ستظهر لك رسالة تستعلم عما إذا كنت تريد إجراء هذه التعديلات على المجلد فقط، أم على المجلد وكل محتوياته.. اختر الأخيرة.

2- برغم أن باستطاعتك تجريب البرنامج الجاهز مباشرة، إلا إنّ الأفضل لك أن تنشئه خطوةً بخطوةٍ بنفسك، متابعًا الشرح في هذا المقطع.

ابدأ مشروعًا جديدًا، مستخدمًا الاسم الافتراضيّ Windows-Application1، وضع مربع نص TextBox وزرًّا Button على النموذج.. استخدم الفأرة لتغيير حجم وموضع الأداتين ليصبحا كما في الشكل التالي.

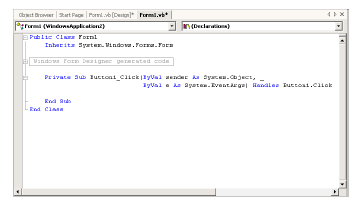


والآن علينا أن نكتب بعض الكود لتقييم لغة المستخدم المفضلة.

**نافذة محرر الكود Code Editor Window:**

إن تطبيقات الويندوز تتكون من أجزاء صغيرة من الأوامر تسمّى "مجيبات الأحداث"Event Handlers، هي التي تتفاعل مع الأحداث المختلفة، التي تنتج عن المستخدم أو النظام.. ففي حالتنا مثلا، نريد أن نبرمج الفعل الناتج عن ضغط المستخدم للزر، ليتم تنفيذ بعض الأوامر التي تعرض رسالة ما.

ولكتابة الكود الذي يتحكم في أداء الزر، انقر مرتين بالفأرة عليه.. ستجد أن نافذة محرر الكود قد ظهرت:



وبها مكتوب ما يلي:

**Private Sub Button1\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**End Sub**

لقد فتح المحرر "الإجراءَ الفرعيَّ ضغط الزر1" Sub Button1\_Click، وهو الذي يجيبُ الحدث الناتج عن ضغط الزر.

تلاحظ أن السطر الأول طويل جدا، ولكي تتمكن من رؤيته على الشاشة كاملا دون أن تضطر لاستخدام المنزلق الأفقي Horizontal ScrollBaar، فإن VB.NET يمنحك الرمز " \_" الذي هو عبارة عن مسافة وشرطة منخفضة Underscore، والذي بإضافته لنهاية السطر، يعتبر VB.NET أن السطر لم ينتهِ بعد، وأن السطر الذي يليله هو مكمل له.. وبهذا يمكنك أن تشطر الأسطر الطويلة إلى عدة أسطر، دون أن تتسبب في خطإ في قواعد اللغة Syntax.

**ملحوظتان:**

1- على عكس لغات البرمجة الأخرى، فإن VB لا يضع علامة خاصة تدل على نهاية الأمر.. لهذا تعتبر اللغاتُ الأخرى أي عددٍ من الأسطر أمرًا واحدًا حتى ينتهي أحد السطور بعلامة التوقف (الفاصلة المنقوطة **";"** في باسكال و C++ مثلا).. بينما كل سطر فيVB يعتبر أمرا مستقلا بمفرده.. فإذا أردت أن تقسم الأمر على سطرين، يمكنك استخدام رمز استمرار السطر Line-continuation character  **" \_"**.. وهناك إمكانية بديلة للتسهيل، حيث تستطيع أن تفعّل ذلك باستخدام خاصية "التفاف الأسطر" word wrap الخاصة بمحرر الكود، وسنتطرق إلى ذلك لاحقا.. أمّا إذا أردت أن تكتب أكثر من أمر في نفس السطر، فافصل بينها بالعلامة **":"**.

2- وتلاحظ أيضا أنّ النصف الثاني من السطر الأول ذو مسافة بادئة أطول من النصف الأول.. هذا حتى يتيسر على المبرمج بمجرد النظر، أن يكتشف أنه ليس سطرا عاديا، وأنه تكملة لسابقه.. إن نقطةً جوهريةً يجب أن تعيها، هي أهمية تنظيم سطور الكود، لأن البرنامج بعد أن يتضخم وتتعقد أوامره، يصبح من العسير قراءته وفهمه ومراجعته لاكتشاف ما به من أخطاء، ما لم يكن منظمًا وغنيا بالتعليقات التي تشرح كل الملاحظات والتفاصيل.. عامة ستجد أن VS.NET يقوم بتنظيم مقاطع الأوامر Code Blocks تلقائيًا، بل ويكتب لك نهايات الجمل بمجرد كتابة بداياتها، فمثلا لو كتبت:

**If x = 0 Then**

وضغطت زر الإدخال، فستجد أن جملة:

**End If**

قد كتبت لك تلقائيا، وأن مؤشر الكتابة قد وضع في السطر التالي لجملة الشرط، وبعد مسافة ابتدائية مناسبة.. إن هذه من السمات الجديدة المريحة في بيئة التطوير، والتي يمكنك أن توقف عملها لو أردت من خيارات محرر الكود ـ وإن كنت لا أنصحك بهذا على الإطلاق!

في أعلى الشاشة ترى الآن شريطين، يتخذان اسمي النموذج، وهما: شريط "تصميم نموذج1" Form1.vb [Design]، وشريط "نموذج1" Form1.vb.

فأما الشريط الأول، فيشير إلى مصمم نماذج ويندوز، هناك حيثُ يمكنك تصميم واجهة البرنامج.. وأما الشريط الثاني، فيشير إلى محرر الكود Code Editor، الذي تكتبُ فيه الكود الذي تعمل به الواجهة.

وفي قمة محرر الكود، يوجد اثنان من مربعات الإدخال أو الاختيار المركبة ComboBoxes، وسنسميها للاختصار القائمة المركّبة، لأنها عبارة عن قائمة ListBox بالإضافة إلى مربع نص TextBox.. هاتان القائمتان المركّبتان هما:

1- قائمة العناصر Objects List: على اليسار، وهي تحتوي على أسماء الأدوات والعناصر الموجودة على النموذج.

2- قائمة الأحداث Events List: على اليمين، وهي تحتوي على أسماء الأحداث الخاصة بكل أداة.

وعندما تختار أداةً أو كائنا من القائمة المركّبة اليسرى، يتم ضبط محتويات القائمة المركّبة اليمنى تبعًا لذلك، لتعرض الأحداث الخاصة بهذه الأداةِ أو هذا الكائن.

ولكي تبرمج حدثا ما لأداة ما، فما عليك إلا أن تختار اسم الأداة من القائمة المركبة على اليسار، واسم الحدث من القائمة المركبة على اليمين.

ونظرًا لأن حدث "الضغط" Click Event، هو الحدث الافتراضيّ للأزرار، لذلك كتب لك محرر الكود مجيب هذا الحدث تلقائيًّا بمجرد النقر المزدوج على الزر.

**"مجيبات الأحداث"Event Handlers :**

إن مجيب الحدث Event Handler هو إجراءٌ فرعيٌّ يتمّ استدعاؤه تلقائيًّا في كل مرة يحدث فيها الحدث.. فمثلا في حالتنا هذه، الحدثُ هو ضغط الزر، لهذا، في كل مرة يضغط فيها المستخدم هذا الزر، يتم استدعاء الإجراء Button1\_Click.. وما عليك الآن سوى كتابة الأوامر في هذا الإجراء، لتتفاعل مع حدث ضغط الزر.

ويتكون اسم الإجراء الذي يستجيب للحدث، من اسم الأداة واسم الحدث تفصلهما شرطة منخفضة Underscore "\_" .. ولكن هذا فحسب مجرد اسم افتراضيّ، ويمكنك أن تغيره إلى أي اسم آخر.

إن ذلك يعود إلى أن الاسم ليس هو ما يحدد كون الإجراء مجيبًا للحدث ـ وهو الأمر الذي عهده مبرمجو الإصدارات السابقة من VB.. إن ما يجعل الإجراء مجيبا لحدث ما، هو الكلمة الأساسية "يجيب" Handles، والتي تأتي في نهاية تعريف الإجراء، متبوعةً باسم الحدث الذي سيستجيبُ الإجراء له، وهو في هذا المثال الحدث Button1.Click.

إن هذه الإمكانية الجديدة في VS.NET لا تمنح فقط المرونة في تسمية مجيب الحدث، ولكنها تسمح كذلك بأن يكون إجراء واحد هو المسئول عن الاستجابة لأكثر من حدثٍ ولأدواتٍ مختلفة، بحيث تؤدي كلها نفس الوظيفة.. وانظر للصيغة المحتملة التالية:

**Private Sub Button1\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click, \_**

**Text1.DoubleClick**

**End Sub**

الشرط الوحيد الذي يجب أن تراعيه، هو أن تكون الأحداث التي يستجيب الإجراء لها متناسبة، بحيث تتشابه معاملاتها، وإلا فسيعترض VB.

ويمكنك في أيِّ وقتٍ أن تغيّر اسم الأداة من خاصية "الاسم" Name في نافذة الخصائص، دونَ أن تقلقَ بخصوص الإجراءات التي تستجيب لها، حيث سيقوم VB.NET بتغيير اسم الحدث المستجاب له في تعبير Handles تلقائيًّا.. وإن كان عليك أن تبحث عن اسم الأداة القديم في الكود لاستبداله بنفسك.

إن كل أداة تتفاعل مع العديد من الأحداث، ومن البدهيّ أننا لن نكتب إجراءاتٍ تستجيب لكل الأحداث الممكنة لجميع الأدوات.. إننا نكتب منها فقط ما تتطلبه وظيفة البرنامج.

كما أن كل أداةٍ تنفذ تلقائيا الأحداث الخاصة بوظيفتها.. فمربع النص يستجيب لضربات أزرار لوحة المفاتيح فيقوم بتعديل محتوياته، وخانة الاختيار CheckBox تغيّر حالتها بوضع أو إزالةِ علامة الاختيار في كل مرة تنقرها بالفأرةِ أو تضغط مسطرة المسافات... وهكذا.

وبفضل هذه الاستجابة التلقائية، فإنك لا تحتاج لكتابة أي كودٍ لتنفيذها، وهو ما جعل اللغات المرئيّة بيئةً سهلةً وسريعةً لإنشاء تطبيقاتٍ قويةً وقياسيّةً بأسرعِ وقتٍ وأقل مجهود.

ولكن هذا لا يمنعك من كتابة الأوامرِ التي تحسّن أو تزيد أداء أدواتك.

**أوّل ما تكتبه من البرمجة:**

والآن حان وقت كتابة بعض الأوامر في الإجراء المستجيب لحدث ضغط الزر.

إن البرمجة تتمّ على خطوات:

1- أن تصف الهدف جيدًا، حتى تستطيع أن تصل إلى الطريقة التي تحقّقه بها.

2- أن تكتب الأوامر التي تنفذ هذه الخطوات، وهو أسهل وأبسط جزء في الموضوع.

3- مراجعة البرنامج واختباره جيدًا للتأكّد من خلوّه من الأخطاء.

إن إبداع المبرمج لا يتعلقُ بكتابة الأوامر والدوال، بقدر ما يتعلق بابتكار حلولٍ شافية للمشاكل البرمجية التي تواجهه، وهنا تظهر أهمية ثقافته وخبرته في المجالاتِ المختلفة، فالبرمجة لا يحدها تخصص.. إنها تتعامل مع كل الأنشطة الذهنية والتجارية والفنية.. لهذا يحتاجُ المبرمج للمرونة الذهنية والقدرة على التعلم المستمر، حتى يستطيع التعامل مع مجالاتٍ قد تكون خارج نطاق دراسته الأكاديمية.

ما علينا.. دعنا نرى الآن ما هو المطلوب من برنامجنا وكيف سنحققه:

1- نحتاج أن نقرأ النص الذي أدخله المستخدم في مربع النص.

2- ثم سنرى إن كان هذا النص هو عبارة "Visual Basic" أم لا:

* 1. فإذا كانت الإجابة هي نعم، فسنعرض رسالة للمستخدم تقول: "اختيار ممتاز".
  2. وإذا كانت الإجابة لا، فسنعرض رسالة تقول: "اختيار جيد".

إن الخطوات السابقة تسمّى خوارزمية Algorithm، نسبةً للعالمِ الرياضيّ العربيّ الشهير (الخوارزمي)، وبكتابتها نكون قد أنجزنا الجزء الأكبر من العمل.. بقيَ أن نكتب الأوامر التي تحقق هذه الخوارزمية.

هذا هو الإجراء المطلوب ـ مع ملاحظة أننا غيرنا اسمه من Button1\_Click إلى "تقييم اللغة" EvaluateLanguage:

**Private Sub EvaluateLanguage\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**Dim Language As String**

**Language = TextBox1.Text**

**If Language = "Visual Basic" Then**

**MsgBox ("اختيار ممتاز")**

**Else**

**MsgBox ("اختيار جيد")**

**End If**

**End Sub**

وهذا هو شرح كيف يُطبّق الإجراء خوارزميتنا:

1- الخطوة الأولى، هي أن نقرأ محتويات مربع النص، وهي تتم على مرحلتين:

أ. تعريف متغير Declaring a variable:

لكي نقرأ محتويات مربع النص فلا بد لنا من مخزنٍ نضعها فيه.. فأين يكون هذا المخزن؟

إن أنسب مكانٍ هو الذاكرة المؤقتة RAM، فهي أسرع من القرص الصلب في التعامل مع البيانات، وهو ما يناسب السرعة التي يتطلبها تنفيذ البرامج.

في بدايةِ عهد البرمجةِ، كانت لغات البرمجة منخفضة المستوى Low Level، وكانت البرمجة تتطلب التعامل مع أجزاء الجهاز المادية Hardware مباشرة، مما كان يشكّل صعوبةً بالغةً على المبرمج.. فمثلا للكتابة في الذاكرة المؤقتةِ، كنا نحتاج لمعرفة كيفية تنظيمها وعناوينها الفيزيائية التي يتعامل معها الجهاز كهربيا.

ثم جاءت لغات البرمجة عالية المستوى High Level، فعزلت المبرمج عن مكونات الجهاز المادية، ومنحته العديد من الأوامر السهلة التي تمكّنه من تنفيذ ما يريده بسرعةٍ وكفاءة.

والآن لحجز مخزنٍ في الذاكرة، كل ما عليك هو أن تعرّف متغيّرًا Variable، سيكون هو موقعك في الذاكرة، حيث يمكنك أن تكتب فيه وتقرأ منه، دون أن تحتاج لمعرفة عنوانه الفيزيائيّ.. كل المطلوب منك هو أن تعطيه اسما مشتقًا من وظيفته في برنامجك ليسهل عليك فهمه وتذكره، وتحدد نوع البيانات التي تريد وضعها فيه.

وفي حالتنا هذه، نحن نحتاج لمتغير نصّيّ String، حتى نضع فيه محتويات مربع النص.. لعلك فهمت الآن وظيفة هذا السطر:

**Dim Language As String**

إن كلمة "عرّف" Dim هي من كلمات لغةVB الأساسية، وهي تطلب من مترجم الأوامر Compiler حجز مكان في الذاكرة، وتسميته Language، وتحدد نوعه بأنه متغير نصّي String.

وجملة التعريف تأخذ الصيغة العامة:

**Dim اسم\_المتغير AS نوع\_المتغير**

حيث اسم\_المتغير هو أي اسم تختاره.. وكما رأيت، أسمينا المتغير Language، لأننا سنخزن فيه اسم لغة البرمجة التي سيدخلها المستخدم في مربع النص.. ستتضح لك أهمية ذلك حينما تكتب برامجك الضخمة بإذن الله، وتجد أنها تحتوي على آلاف المتغيرات، مما سيجعل من المستحيل عليك إعادة فهم برنامجك لتطويره أو لتعديل خطإٍ فيه، ما لم تكن كل أسماء المتغيرات مفهومةً ودالةً على وظيفتها.

ورغم أن VB لغة غير حساسة لحالة الحرف ـ أي يمكنك أن تكتب الكلمات دون الاهتمام بالحروف الكبيرة Capital والصغيرة Small ـ إلا أننا كتبنا أول حرفٍ من اسم المتغير كبيرا.. والآن لو كتبت كلمة language بحرف l صغير، فستجد بمجرد تركك للسطر الذي كتبتها فيه، أن VB قد حولها تلقائيًّا إلى Language.

بقيَ أن نوضح نوع\_المتغير.. إن هناك العديد من الأنواع، مثل العدد الصحيح Integer، والعدد الطويل Long، والنص String.. وسنتطرق لذلك في حينه حين نحتاجه.

ب. وضع قيمة في المتغير Assigning a value to a variable:

بعد أن عرفنا المتغير، يمكننا الآن أن نضع فيه قيمة خاصية Text الخاصة بمربع النص، وذلك بعملية تساوٍ بسيطة.

**Language = TextBox1.Text**

ما يجب أن تلاحظه هنا، هو أن المتغير النصيّ مرن جدا، حيث يمكنك أن تضع فيه أي نصٍّ بأي طول، وهو سيتمكن من التمدد أو الانكماش لاستيعاب حروف نصك.

2- الخطوةُ الثانية هي مقارنة قيمة المتغير Language مع النص "Visual Basic".. ولفعل ذلك نكتب جملة شرطية If statement لتختبر تساوي القيمين:

**If Language = "Visual Basic" Then**

أ. فإذا كان الشرط صحيحًا True، فسينفذ البرنامج السطر:

**MsgBox ("اختيار ممتاز")**

و MsgBox هي إحدى دوال VB الأساسية، وهي تقوم بعرض رسالة للمستخدم ـ تلك التي تكتبها بين القوسين كمعاملٍ للدالة.. وتظهر الرسالة على الشاشة كنافذة صغيرة، عليها كلماتك وزر "موافق" OK، والذي يجب على المستخدم أن يضغطه لإغلاق الرسالة.

لعلك الآن استنتجت أن المقطع Msg في اسم الدالة هو اختصار للكلمة الإنجليزية Message بمعنى رسالة.. ولعلك لاحظت أن كلا من مقطعي الكلمة يبدأ بحرف كبير، وذلك حتى يسهل على العين تمييز المقطعين، وبالتالي يمكن قراءة الكلمة المركّبة.

وبعد تنفيذ هذا السطر، سيتجاوز البرنامج مقطع "غير ذلك" Else بما يحويه من جمل، وسيقفز مباشرةً لسطر نهاية الجملة الشرطية End If.

ب. أمّا إذا كان الشرط خاطئًا، فسيتجاوز البرنامج مقطع الشرط، وسيقفز مباشرةً لجملة Else، حيث سينفذ الجملة:

**MsgBox ("اختيار جيد")**

التي تفهم الآن وظيفتها.

**ملاحظة 1:**

يمكن كتابة نفس جملة استدعاء الرسالة على الصيغة التالية:

**Call MsgBox ("اختيار جيد")**

وكلتا الصيغتين متكافئة، وإن كانت الأولى مختصرةً، والثانية أكثر وضوحًا لك عند إعادة قراءتها فيما بعد.

**ملاحظة 2 (لمبرمجي VB6):**

في الإصدارات السابقة من VB، كان لا بد ألا تضع القوسين، في حالة استدعاء دالةٍ دونَ أن يسبق اسمها كلمة الاستدعاء Call.. لقد تغير الوضع الآن، وصار عليك في كل الأحوالِ أن تحيط المعاملات التي ترسلها للدالة بقوسين، سواء كتبت كلمة الاستدعاء أم لا.

**ملاحظة 3:**

لعلك لاحظت أن الأوامر تكتبُ في مقاطع Blocks، كل منها لها بادئة وخاتمة:

- فمقطع الإجراء Procedure يُكتبُ على الشكل:

**Sub اسم\_الإجراء(معاملات\_الإجراء)**

**أوامر\_الإجراء**

**End Sub**

- ومقطع الدالة Function يُكتبُ على الشكل:

**Function اسم\_الدالة(معاملات\_الدالة) As نوع\_المتغير**

**أوامر\_الدالة**

**End Function**

- ومقطع جملة الشرط يكتبُ على الشكل:

**If شرط Then**

**[أي أوامر بما فيها مقاطع الأوامر ومقاطع الشرط نفسها.]**

**ElseIf شرط Then**

**[أي أوامر بما فيها مقاطع الأوامر ومقاطع الشرط نفسها.]**

**[مع إمكانية تكرار جملة ElseIf لأي عدد من المرات.]**

**Else**

**[أي أوامر بما فيها مقاطع الأوامر ومقاطع الشرط نفسها.]**

**End If**

وغير ذلك من مقاطع الأوامر، والتي سنشرحها كلها بالتفصيل فيما يلي من فصول.

**تحسين التطبيق ليصبح أفضلَ أداءً:**

ماذا سيحدث لو أخطأ المستخدم في إدخال جملة "Visual Basic"؟

في هذه الحالة، سيظن البرنامج أن المستخدم قد أدخل اسم لغةٍ أخرى!

ليس هذا فحسب، بل إن مجرد إدخال المستخدم لأحد الحروف في حالة غير حالتها ـ مثل أن يكتب "visual Basic" بحرف v صغير ـ سيؤدي إلى نفس النتيجة السابقة!

وعلاجُ المشكلةِ الأخيرةِ أسهل، فكل ما نحتاج إليه هو إجراء المقارنة بين الكلمات وكل حروفها لها نفس الحالة.. فمثلا يمكننا أن نحول ما يدخله المستخدم إلى الأحرفِ الكبيرة Uppercase، ونقارنه بجملة "VISUAL BASIC".

و تمنحك VB.NET دالةً جاهزةً لأداء ذلك، تنتمي لخليّة النصوص String Class، وهي الدالة "حوّل للحالة العليا" ToUpper.

**ملحوظة هامة:**

VB.NET هي لغة برمجة مبنية بالكامل على الكائنات Objects، لدرجة أن أنواع المتغيرات الرقمية والنصية قد صارت خلايا Classes، على غير ما كانت في VB6.. ولكي تدرك الفارق، جرب أن تكتب الجملة التالية في نهاية إجرائنا:

**Language = String.**

مبدئيًّا ستلاحظ أن محرر الكود قد أظهر لك قائمةً تحتوي على مجموعةٍ من الأسماء، بمجرد كتابة النقطة التي تلي كلمة String.. ما تراه في القائمة هو أسماء الخصائص Properties والوسائل Methods، التي تنتمي لخليّة النصوص String Class.. وسنتعرف بالتفصيل على هذه الخليّة فيما بعد.

ليس هذا كل ما في الأمر، بل إن أي متغير يتم تعريفه، هو في حقيقة الأمر نسخة من الخليّة التي ينتمي إليها نوع المتغير، بمعني أن هذا المتغير يرث كل خصائص ووسائل هذه الخليّة.. جرب أن تكتب السطر التالي:

**Language = language.ToUpper**

إن المتغير "Language" قد ورث الدالة ToUpper، والتي سيتم تطبيقها على النص الذي يحتويه هذا المتغير، لإنتاج نص جديد كل أحرفه كبيرة.. هذا ما حدث على يمين علامة التساوي.. الآن يتم تنفيذ عملية التساوي، حيث يوضع النص الناتج من الدالة في المتغير نفسه، لاستبدال القيمة القديمة.. ويمكننا بالطبع أن نضع النص الناتج في أي متغيرٍ آخر، كالتالي:

**Dim UpperCaseLanguage As String = language.ToUpper**

وعلينا أن نفترضَ أيضًا أن المستخدم قد يريد إدخال اسم اللغة مختصرًا كأن يكتب "VB" أو "VB.NET".. تعال إذن نعدل الإجراء ليصير كالتالي:

**Private Sub EvaluateLanguage\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**Dim language As String**

**language = TextBox1.Text**

**language = language.ToUpper**

**If language = "VISUAL BASIC" Or language = "VB" Or \_**

**language = "VB.NET" Then**

**MsgBox("اختيار ممتاز")**

**Else**

**MsgBox("اختيار جيد")**

**End If**

**End Sub**

تلاحظُ في هذا المثال أنّنا كتبنا جملة شرطٍ مركّبة، فاستخدمنا المعامل "أو" Or Operator، لنتساءل عن حدوث واحدٍ على الأقل من الشروط المطلوية.

ويمكن استخدام المعامل "و" And Operator، لنتساءل عن حدوث جميع الشروط المطلوبة معا في نفس الوقت.

ويمكن بالطبع المزج بين "أو" و "و" في نفس جملة الشرط لتركيب جمل معقّدة، على حسب احتياجنا.. وسنرى ذلك بالتفصيل فيما يلي من فصول.

والآن شغل البرنامج، وجرب هذه الإمكانيات الجديدة.

أرجو أن تكون قد لمست مدى السهولة والبساطة التين صممتَ بهما هذا البرنامج.. ستجد كل برامجك بهذه البساطة إن شاء الله، فإن VB.NET تمدّك بثروةٍ من الأدوات والدوال، التي تجعل عملك غايةً في الراحة.

وما زال لدينا المزيد لتطوير برنامجنا.. فمن الأذكى أن نحد من قدرة المستخدم على كتابة كل ما يريد كمدخل للبرنامج، وذلك حتى نحدّ من احتمال خطئه في الإدخال.. وفي حالتنا الراهنة، يمكننا أن نعرض للمستخدم أسماء لغات البرمجة المتاحة، وكل ما عليه هو أن يختار لغته من بينها.. واحدة من الطرقِ لفعلِ ذلك، هي استخدام القائمة المركّبة ComboBox.. هذا هو موضوع الفقرة التالية.

**جعل تطبيقنا أكثر حميميةً للمستخدم More User-Friendly:**

ابدأ مشروعا جديدًا وأسْمِه WindowsApplication2.. عندما يظهر لك النموذج في بيئة التطوير، حدد خاصية "الخط" Font في نافذة الخصائص، واضغط زر الانتقال Ellipsis Buttonلتعرض مربع حوار الخط، حيث يمكنك اختيار الخط وخصائصه لهذا النموذج.. اختر خط Comic Sans MS 11 نقطة.. بهذا سترث أي أداة تضعها على النموذج هذا الخط ومواصفاته.

افتح صندوق الأدوات Toolbox، وانقر مرتين على أيقونة القائمة المركّبة ComboBox، لتضيف واحدةً منها إلى النموذج.. أضف إلى نموذجك كذلك زرًّا، وحاول أن تضبط وضع الأداتين ليبدو النموذج كما في الشكل.

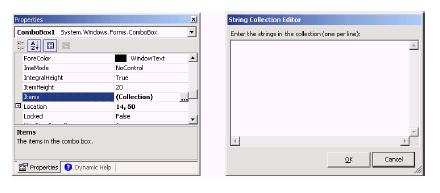


اكتب النص "تقييم اختياري" في خاصية Text الخاصة بالزر.

**ملحوظة:**

عند كتابة الكود، يجب وضع النصوص داخل علامتي تنصيص، تمييزًا لها عن أسماء المتغيرات.. ولكن في نافذة الخصائص، لا تكتب النصوص بين علامتيْ تنصيص، إلا إذا أردت أن تظهرا كحرفين ضمن النص نفسه.

علينا الآن أن نملأ القائمة المركّبة بعناصر الاختيار.. اضغط القائمة المركّبة بالفأرة مرةً، وحدد لها خاصية "العناصر" Items في نافذة الخصائص.. وهذه الخاصية هي تجمّع Collection، بمعنى أنها تستطيع أن تأخذ مجموعةً من القيم النصيّة بدلا من قيمة واحدة.. اضغط زر الانتقال، ليظهر لك مربع حوار محرر نصوص التجمّع.



وهو عبارة عن مربع نص، يمكنك أن تكتب فيه العناصر التي تريدها أن تظهر في القائمة المركّبة.... اكتب العناصر التالية، كل عنصرٍ في سطر مستقل:

**C++**

**C#**

**Java**

**Visual Basic**

**Cobol**

ثم اضغط نعم لإغلاق مربع الحوار.

لن تظهر العناصر في القائمة في وقت التصميم، ولكنك ستراها عندما تشغل البرنامج.

حدد خاصية Text الخاصة بالقائمة المركّبة، وضع فيها النص: "اختر لغتك المفضلة".. لن يكون هذا النص أحد عناصر القائمة، ولكنه سيكون أول ما سيظهر عند تشغيل البرنامج، داخل مربع النص الخاص بالقائمة المركّبة.

شغّل البرنامج، وانظر أداء القائمة المركبة.. ستجد أنها تؤدي تمامًا نفس المهام التي عهدتَها في باقي تطبيقات الويندوز.

اضغط زر إسدال القائمة، واختر أي عنصرٍ بغطه بالفأرة.. أو اضغط سهمَيِ الأسفل والأعلى من لوحة المفاتيح لإظهار العنصر التالي أو السابق من القائمة دون أن تنسدل.

ما زلنا لم نخبر البرنامج ماذا سيحدث عند ضغط الزر.. أغلق البرنامج وتعال نضيف بعض الكود إليه.

انقر مرتين على الزر لتظهر لك نافذة الكود، حيث ستعرض لك حدث ضغط الزر.. اكتب داخله الأوامر التالية:

**Private Sub EvaluateLanguage\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**Dim language As String**

**language = ComboBox1.Text**

**If language = "Visual Basic" Then**

**MsgBox("اختيار ممتاز")**

**Else**

**MsgBox("اختيار جيد")**

**End If**

**End Sub**

تذكر أن القائمة المركّبة ستعرض جملة "اختر لغتك المفضلة" عند تشغيل البرنامج.. وهذه الجملة لا تُشير إلى اسم لغة برمجية، ولن يكون من الصواب أن يضغط المستخدم الزر وهي معروضة في القائمة.. يمكن تلافي ذلك، عن طريق جعل القائمة المركبة تعرض اسم إحدى اللغات عند تشغيل البرنامج.

إن عرض النموذج على الشاشة يحتاج أولا لتحميله في الذاكرة.. وعند تحميل النموذج، يتم إطلاق الحدث "تحميل" Load Event.

انقر مرتين بالفأرة في أي موضعٍ خالٍ من النموذج.. سيظهر لك حدث تحميل النموذج.. أضف له هذا الكود، لاختيار العنصر من القائمة عند تحميل النموذج:

**Private Sub Form1\_Load(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles MyBase.Load**

**ComboBox1.SelectedIndex = 3**

**End Sub**

وبما أننا قد اخترنا أحد العناصر من الكود، فلا أهمية للنص المكتوب في خاصية Text للقائمة المشتركة.. يمكنك أن تمسحه الآن!

لعلك الآن قد أدركت أن الأدوات الموجودة في صندوق الأدوات، هي أكثر من كونها مجرد رسومات توضع على النماذج.. إنها تحتوي على العديد من الخصائص والوسائل، التي تؤّدي العديد من الوظائف.

**مكونات بيئة التطوير**

إنه لمن العسير أن نشرح كل مكونات بيئة التطوير دفعةً واحدة ـ لهذا سنكتفي هنا بشرح الأجزاء الرئيسية التي سنستخدمها في الفصول الأولى من هذا الكتاب:

* [**القوائم الرئيسية Menus:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\4-%20مكونات%20بيئة%20التطوير\1-%20القوائم%20الرئيسية.htm)
* [**أهمّ نوافذ بيئة التطوير:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\4-%20مكونات%20بيئة%20التطوير\2-%20أهمّ%20نوافذ%20بيئة%20التطوير.htm)
* [**خيارات بيئة التطوير Environment Options:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\4-%20مكونات%20بيئة%20التطوير\3-%20خيارات%20بيئة%20التطوير.htm)

**القوائم الرئيسية Menus**

* + [**ما هي:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\0-%20ما%20هي.htm)
  + [**القائمة الرئيسية "ملف" File Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\0-%20ملف.htm)
  + [**القائمة الرئيسية "تحرير" Edit Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\1-%20تحرير.htm)
  + [**القائمة الرئيسية "عرض" View Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\2-%20عرض.htm)
  + [**القائمة الرئيسية "مشروع" Project Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\3-%20مشروع.htm)
  + [**القائمة الرئيسية "بناء" Build Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\4-%20بناء.htm)
  + [**القائمة الرئيسية "تصحيح" Debug Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\5-%20تصحيح.htm)
  + [**القائمة الرئيسية "بيانات" Data Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\6-%20بيانات.htm)
  + [**القائمة الرئيسية "تنسيق" Format Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\7-%20تنسيق.htm)
  + [**القائمة الرئيسية "أدوات" Tools Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\8-%20أدوات.htm)
  + [**القائمة الرئيسية "نافذة" Window Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\9-%20نافذة.htm)
  + [**القائمة الرئيسية "استعلام (مساعدة)" Help Menu:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\1-%20القوائم%20الرئيسية\10-%20استعلام.htm)

**ما هي؟:**

القوائم الرئيسية هي الأسماء التي تراها في أعلى الشاشة الرئيسية لبيئة التطوير، مثل File و Edit، وتحتوي كل منها على العديد من الأوامر Commands والقوائم الفرعية Submenus.

وليست كل الخيارات والقوائم الفرعية مرئيةً Visibleأو فعّالةً Enabledطوال الوقت، فذلك يتوقف على إذا ما كان من الممكن تطبيقها على بيئة التطوير في اللحظة الراهنةِ أم لا.. فمثلا تختفي قائمة "البيانات"Menu Data، إذا انتقلتَ لمحرر الكود، فأنت ببساطة لن تستطيع استخدام إمكانياتها بينما تكتب الكود.

فلنشرح الآن بعض القوائم الرئيسية.

**القائمة الرئيسية "ملف" File Menu:**

وتحتوي على الأوامر الخاصة بفتح وتخزين الملفات والمشاريع ومكونات المشاريع، بالإضافة للأوامر الخاصة بإضافة عناصر ـ جديدة أو موجودة على الجهاز ـ للمشروع الحالي.

**القائمة الرئيسية "تحرير" Edit Menu:**

وتحتوي على الأوامر المعتادة، الخاصة بتحرير النصوص والعناصر، مثل النسخ والقص واللصق والحذف والبحث والاستبدال.. كما أنها تحتوي على أوامر متقدمة مثل:

**القائمة الفرعية "متطور" Advanced:**

وهي لا تظهر إلا حينما تكون في محرر الكود.. وهي تحتوي على الأوامر التالية:

**الأمر "أظهر المسافات الفارغة" View White Space:**

يؤدي هذا الأمر إلى استبدال المسافات برموز خاصة.. إن المسافات مهمة جدا في تنظيم سطور الكود ومقاطع الأوامر، حتى يسهل قراءة الكود فيما بعد.

**الأمر "التفاف الأسطر" Word Wrap:**

يؤدي لالتفاف السطر تلقائيا إلى سطر جديد، إذا تجاوز طوله عرض الشاشة، حتى لا يضطر المبرمج إلى استخدام المنزلق الأفقي لقراءة بقية السطر.. واختيار هذه الخاصيّة، سيريحُك من كسر الأسطر الطويلة إلى سطورٍ أصغر باستخدام الشرطة المنخفضة "\_ " Underscore.

**الأمر "اعتبر / لا تعتبر التحديد تعليقا" Comment Selection / Uncomment Selection:**

التعليقات هي جمل تضعها في وسط الكود لتشرح شيئًا أو لتلفت النظر إلى شيء.. هذه التعليقات يتجاهلها مترجم الأوامر عند تنفيذ البرنامج.. وأحيانا ما نحتاج إلى تهميش بعض الأسطر من الكود حتى لا ينفذها البرنامج، ولكن دون حذفها، حتى يمكن استعادتها إذا ما احتجنا إليها.. ولكي نحول سطرًا إلى تعليق، لا نفعل أكثر من وضع علامة التعليق " '" أو كتابة كلمة "Rem:" أمام السطر.. مثال:

**' هذا تعليق.. لاحظ العلامة في بداية السطر**

**Rem: هذا أيضا تعليق، وإن كانت الطريقة الأولى أسرع وأسهل**

وللوضوحِ البصريّ، يمنح محرر الكود سطرَ التعليقِ لونًا مختلفًا ـ أخضرَ في الوضع التلقائيّ.. ولإعادة السطر عاديا مرة أخرى، عليك فقط أن تحذف علامة التعليق.

ولكن الأمر سيكون شاقًا لو أردت أن تحول إلى تعليق، عشرين سطرًا ـ مثلا ـ دفعة واحدة.. هنا يمكنك تحديد كل السطور ثم استخدام أمر "اعتبر / لا تعتبر التحديد تعليقا" ليحولها إلى تعليق.. ثم لو ضغطته مرة ثانية فسيزيل من السطور كلها علامة التعليق.. هكذا بدون أية مشقة!

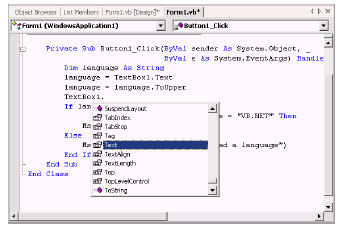
**القائمة الفرعية "المساعدة الذكية" IntelliSense Submenu:**

ووظيفتها تسهيل كتابة الأوامر على المبرمج، لتقليل احتياجه لحفظ الصيغ والهجاء.. وهذه الإمكانيات من أبدع ما سيصادفك في بيئة التطوير، والتي ستشعرك بالارتياح، وتوفر لك الكثير من الوقت والجهد.

وهي تحتوي على أربعة أوامر:

**الأمر "إظهار قائمة الأعضاء" List Members:**

عندما تُفعّلُ هذا الأمر، يحدث ما ذكرناه سابقًا، من ظهور قائمة على الشاشة، تحتوي على كل الخصائص والوسائل والأحداث التابعة للكائن الحالي ـ مثل الكائن النصّي Language، ومثل أسماء الأدوات كـTextBox1 ، و Button1 في البرنامج السابق ـ بمجرد أن تكتب بعد اسمه نقطة "." .



إن هذا يتيح لك أمورا في غاية الأهمية:

- يريحك من عناء تذكر كل أسماء أعضاء هذا الكائن، وهي كثيرة بالفعل لكل كائن، بخلافِ أن عدد الكائنات في اللغة نفسها هائل، ويمكن أن تصادفك كائنات لا توجد عنها معلومات في إرشادات اللغة، لأنها كائنات من خارج اللغة، كتبها مبرمجون هواة، أو شركات أخرى غير ميكروسوفت.. في هذه الحالة، تصبح هذه أسهل طريقة للتعرف على محتويات الكائن وقدراته.

- يسهّل لك الانتقال بين الأعضاء Members بالفأرة أو بلوحة المفاتيح، وفي كل مرة تحدد فيها أحد الأعضاء، يظهر لك تلميح على الشاشة يشرح وظيفة العضو بطريقة مختصرة، وبهذا يمكنك أن تتعلم بسرعةٍ العديد من خصائص ووظائف الكائن، وتستطيع العثور على الوظيفة التي تريدها، حتى لو لم تكن تعرفها من قبل.

- بالضغط المزدوج بالفأرة، أو ضغط المسافة أو زر الإدخال من لوحة المفاتيح، يمكنك كتابة اسم العضو المحدد حاليا في القائمة، في الموضع الحاليّ من الكود، وبهذا توفر وقت كتابة الكلمة بنفسك، وترحم نفسك من كثرة الأخطاء الهجائيّة.

وحتى لا تضطر للتنقل بين الأعضاء طويلا، للوصول إلى عضو متأخر في الترتيب، اكتب حرفًا أو حرفين من اسمه، وستجد أن القائمة تنتقل تلقائيّا لتحدد العضو الذي يبدأ بما كُتب من حروف.. وعندما يظهر لك اسم العضو على الشاشة اضغطه بالفأرة مرتين، أو صِلْ إليه بالأسهم من لوحة المفاتيح واضغط مسافة، عندئذ سيتم إكمال كتابة الاسم في الكود.

وإذا كانت هناك خاصية لا تقبل أيةَ قيمةٍ إلا من بين مجموعةٍ محددةٍ من القيم، ستجد أن قائمةً بهذه القيم ستظهر لك فور كتابتك لعلامة "=" بعد اسم الخاصية.. فلو كتبت:

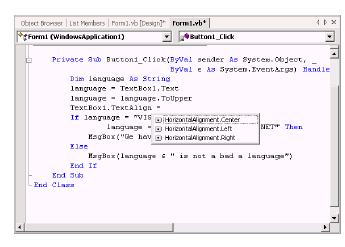
**TextBox1.TextAlign =**

فستظهر لك قائمةً تحتوي على القيم التالية، وهي فقط ـ أو ما يعادلها من أرقام ـ المسموح بها للخاصية:

**HorizontalAlignment.Center**

**HorizontalAlignment.Right**

**HorizontalAlignment.Left**



ويمكنك اختيار القيمة التي تريدها، بالطرق نفسها السابق ذكرُها.

وفي أي وقتٍ تريد فيه إلغاء القائمة، اضغط زر الهروبEsc من لوحة المفاتيح.

**الأمر "معلومات عن المتغيرات" Parameter Info:**

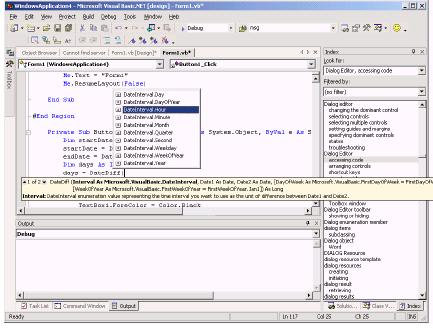
بينما تكتب الكود، يمكنك أن تحرك مؤشر الفأرة فوق أي متغير أو وسيلة (دالة أو إجراء) أو خاصية، وتتركه ثابتا للحظة ـ لا تضغط زر الفأرة، فأنت لا تحتاج لنقل مؤشر الكتابة (العلامة الضوئية) من موضعه.. ستجد أن تلميحًا ظهر على الشاشة، يعرض لك جملة تعريف هذا العنصر Declaration.. سيساعدك هذا على تذكر نوع هذا العنصر.

**الأمر "معلومات عاجلة" Quick Info:**

كثيرًا ما يتوجب عليك أن ترسل بعض المعلومات للخاصية أو الدالة أو الإجراءِ لتتحكم في أدائها.. هذه المعلومات تسمّى معاملات Parameters (Arguments)، وهي توضع بين قوسين بعد اسم الخاصية أو الدالةِ أو الإجراء مباشرة، ويفصل بين كل معاملٍ والآخر علامة ",".

وإلى VB4 كان عليك أن تحفظ متغيرات كل دالة: أنواعها وترتيبها ووظيفتها، وهو أمر غاية في الصعوبة، وكان يضطر المبرمج إلى كثرة الرجوع إلى إرشادات اللغة ليقرأ المعلومات الواردة فيها عن الدالة التي يريد استخدامها ومعاملاتها.

ولكن فيما بعد، منحتنا ميكروسوفت هذه الطريقة السهلة: فبمجرد أن تفتح أول القوسين، يظهر لك تلميح على الشاشة، يعرض لك الجملة التي عرّفت فيها الدالة، بما فيها معاملاتها، بحيث لا تجهد ذهنك في تذكر الأنواع أو الترتيب، مكتفيا بالتركيز في صحة المعلومات التي ترسلها ـ حتى لا تتسبب معلومة ما في خطإٍ برمجيّ.



وفي VS.NET، هناك تسهيل جديد، فعند الانتقال لموضع معاملٍ لكتابة قيمته، يظهر في التلميح شرح مختصر له، ليذكرك بوظيفته، وعند الانتقال إلى معامل آخر ـ بكتابة علامة "," التي تفصل معاملا عن غيره ـ يظهر شرحه في التلميح ، وهكذا.... .

وكمثال، جرب أن تكتب اسم الدالة DateDiff()، وهي تحسب الفرق بين تاريخين في نطاق زمنيٍّ معين، وانظر ماذا سيحدث عندما تكتب القوس الأول.

**الأمر "أكمل الكلمة" Complete Word:**

ولمزيد من التسهيل، يمنحك هذا الأمر القدرة على إكمال الأحرف التي كتبتها من الكلمة، بمجرد ضغط الزرين Ctrl + مسطرة المسافات معًا.. فلو كانت حروفك لا يمكن إكمالها إلا لكلمة واحدة، فستجد أن هذه الكلمة قد كُتبت لك تلقائّيا، ولو كان هناك أكثر من كلمة تبدأ بهذه الحروف، فستظهر لك قائمةً تحتوي أسماء كل عناصر اللغة وعناصر برنامجك، محدَّدا فيها أول كلمة تبدأ بالحروف التي كتبتها.

وكمثال: جرب كتابة الحروف "TextB"، واضغط Ctrl + مسطرة المسافات، وانظر ماذا ترى.

**القائمة الرئيسية "عرض" View Menu:**

وتحتوى على الأوامر التي تعرض أيا من أشرطة الأدوات ونوافذ بيئة التطوير، ولقد سبق لك استخدام القائمة الفرعية "أشرطة الأدوات" من قبل.

أما باقي القوائم الفرعية، فهي لعرض بعض النوافذ الأساسية، مثل نافذة "المخرجات" Output، وهي التي تظهر بها رسائل مترجم الكود Compiler، ونافذة "الأوامر" Command، التي يمكنك أن تكتب بها بعض الأوامر وتنفذها مباشرة ـ على سبيل التجريب.

**القائمة الرئيسية "تنسيق" Format Menu:**

وهي لا تظهر إلا إثناء تصميم نماذج الويندوز أو نماذج الإنترنت، وهي تحتوي على الأوامر الخاصة بمحاذاة الأدوات على النموذج، كما سنرى فيما بعد.

**القائمة الرئيسية "بيانات" Data Menu:**

وتحتوي على الأوامر التي تستخدم في المشاريع التي بها تعامل مع قواعد البيانات، وسنتعرف على ذلك بالتفصيل في حينه بإذن الله.

**القائمة الرئيسية "مشروع" Project Menu:**

وتحتوي على الأوامر الخاصة بإضافة العناصر للمشروع، مثل النماذج والملفات والمكوّنات، بل وحتى مشاريع أخرى!

آخر أمر في هذه القائمة، هو "اجعله المشروع الذي يبدأ به التنفيذ" Set As StartUp Project، وكما هو واضح من اسمه، فهو يوضح للمترجم Compiler ـ في حالة وجود أكثر من مشروع في متصفح المشاريع ـ أيا من هذه البرامج سيتم تشغيله عند ضغط F5.

أما بالنسبة للأمرين "أضف مرجعية" و"أضف مرجعية للإنترنت"، فسنتعامل معهما في فصولٍ لاحقة.. وفي عجالة، فإنهما يستخدمان لربط البرنامج بأدوات ومكونات غير قياسية، حتى يمكن استخدامها في المشروع.

**القائمة الرئيسية "تصحيح" Debug Menu:**

إن كلمة Bug في اللغة الإنجليزية تعني "البق" أو الحشرة.. وكلمة Debug تعني إزالة البق ـ من حشية الفراش مثلا ـ وأصبح هذا المصطلح يرمز لتتبع الأخطاء في البرامج واصطيادها وتصحيحها.. وسنختصر هنا كل هذه الحدوتة بترجمة كلمة Debug إلى "تصحيح الأخطاء" واختصارًا إلى "تصحيح".

منطقيٌّ الآن أن تستنتج أن هذه القائمة الرئيسية تحتوي على الأوامر اللازمة لتصحيح البرنامج، بالإضافة للأوامر اللازمة لتشغيله وإيقافه، فلن يمكنك اكتشاف الأخطاء بالطبع بدون تجريب البرنامج!

**القائمة الرئيسية "بناء" Build Menu:**

وهي تحتوي على الأوامر اللازمة لترجمة Compiling (بناء Building) برنامجك.

إن الكود الذي تكتبه هو مجرد سطور من الكلمات ليس لها أي معنى إلا داخل لغة VB.NET.. لهذا نحتاج لبناء البرنامج، بمعنى أن نحول هذه السطور من مجرد كلمات إلى شفرة يمكن تشغيلها على نظام التشغيل، حتى دون وجود VB.NET على الجهاز.

هذا هو دور المترجم Compiler، الذي يقوم بتحويل سطور الكود من اللغة التي كتب فيها إلى لغة ميكروسوفت الوسيطة Microsoft Intermediate Language (MSIL)، وكتابة الناتج في ملف تنفيذي Executable File له الامتداد (.EXE)، حيث يمكنك تشغيله من خارج اللغة كأي برنامج عادي.

**ملاحظات:**

1- إذا لم تفهم الملاحظات التالية فلا تجزع، فستتضح لك أكثر كلما تقدمت في هذا الكتاب.. ولكن على الأقل جرب قراءتها، لأنها ستعطيك رؤية أشمل لبعض المفاهيم البرمجية.

2- الملف التنفيذي هو الصورة النهائيّة لبرنامجك، وهو الذي توزّعه للمستخدمين.. إنّ لهذا سببين في غاية الأهمّيّة:

- حتّى لا يضطر المستخدم إلى شراء لغة البرمجة لتشغيل البرنامج!

- حتّى لا توزّع الكود الرئيسيّ Source Code الذي تعبت لكتابته.. إن هذا يضمن ألا يعرف أحد كيف كتبت البرنامج.. إنّ الملفّ التنفيذيّ لا يحتوي على الكود الذي كتبته، ولكنّه يحتوي على رموز مكتوبة بلغة ميكروسوفت الوسيطة (MSIL).

3- لكي يعمل البرنامج على جهاز غير جهازك، يجب كتابة برنامج إنزال (تنصيب ـ إعداد) SetUp له، حيث يقوم هذا البرنامج بنقل الملفات التي لا بد من وجودها على الجهاز الآخر لتشغيل برنامجك.. وكذلك لا بد من وجود "إطار العمل" .NET FrameWork، وهو المسئول عن تشغيل البرنامج على نظام التشغيل الموجود، بتحويل لغة ميكروسوفت الوسيطة (MSIL) إلى لغة الآلة Machine Code التي يفهمها نظام التشغيل.

4- لغة الآلة هي عبارة عن مجموعة من الأرقام والرموز، يصعب على الإنسان فهمهما، يرمز كل منها لوظيفة معينة، حيث يقوم نظام التشغيل بإرسالها إلى المشغّل الدقيق Microprocessor، وهو بمثابة مخّ الجهاز، فيقوم هو بتنفيذ هذه الأوامر، بالتعاون مع مكونات الجهاز الأخرى كالذاكرة المؤقتة RAM، وأجهزة الإدخال والإخراج.

5- نظرًا لأن أوامر التشغيل تختلف من مشغّل دقيق لآخر، تبعًا للشركة التي صنعته والجيل الذي ينتمي إليه، ونظرًا للاختلافات الكبيرة في المكونات الماديّة بين الأجهزة، فإن إحدى مهام نظام التشغيل الرئيسية، هي توفير بيئة متطورة للمستخدم والمبرمج، تعزلهما عن هذه الاختلافات، بحيث يؤدي كل منهما عملَه بطريقة مستقرة، دون أن يحتاج لتعلم وظائف مختلفة لكل جهازٍ مختلف.

6- يمكن أن نلخصَ الأمر في أن نظام التشغيل هو إحدى الطبقات التي تفصل بينك وبين مكونات الجهاز المادية، وإن كانت تمنحك الدوال اللازمة لأداء أي وظيفة تريدها من هذه المكونات، بطريقة سهلة ويمكن فهمها ـ مثل دوال API التي تستطيع بها أن تفعل معظم ما يفعله الويندوز نفسه، كقراءة الملفات، والتعامل مع الأقراص والمحركات... إلخ.

7- أحببتُ أن أشرح لك كل هذا، حتى تستنتج أن الملفات التنفيذية والبرامج، تُقرأ بطرقٍ مختلفةٍ على كل نظام تشغيل، ومن هنا يتضح أهمية وجود اللغة الوسيطة MSIL و"إطار العمل" .NET FrameWork، حتى يمكن إنتاج برامج تستطيع العمل على أي نظام تشغيل.

8- إن "إطار العمل" هو طبقة جديدة تعزلك حتى عن نظام التشغيل نفسه، بل ويمنحك نفس القدرات ونفس بيئة التطوير بغض النظر عن لغة البرمجة التي تكتب بها ـ وهناك حاليا أكثر من 40 لغة برمجة تستخدم هذا الإطار.. وسرعانَ ما ستكتشف ـ كمبرمج VS.NET ـ أنك نادرًا ما ستستخدم دوال API، لأن "إطار العمل" يمنحك معظم إمكانياتها ووظائفها بطريقة أكثر سهولةً وأمانًا، حتى يريحك من اعتماد كثير من هذه الدوال على نظام التشغيل الذي تعمل عليه.

وأهم أمرين في قائمة "بناء" Build:

**1- الأمر "بناء" Build:**

وهو يبني البرنامج التنفيذيَّ للمشروع، ولكنه لا يعيد ترجمة أي ملف أو مكون، لم تحدث فيه اختلافات منذ آخر مرة تمت ترجمته فيها.

1. **الأمر "إعادة بناء الكل" Rebuild All:**

وهو يبني البرنامج التنفيذيّ، بترجمة جميع الملفات والمكونات من جديد، حتى لو لم يحدث فيها تغيير منذ آخر عملية ترجمة.

**القائمة الرئيسية "أدوات" Tools Menu:**

وتحتوي على العديد من الأدوات، التي تستخدم معظمها في لغة C++.

ومن ضمن الأدوات المتاحة، الوحدات النمطية Macros، وهي تمنحك نفس القدرة المتاحة في تطبيقات Office، على أن تبسّط وظائف بيئة التطوير، بوضع المهام كثيرة التكرار في وحدة نمطية تنفذها آليا.

**القائمة الرئيسية "نافذة" Window Menu:**

وهي قائمة قياسية تجدها في أي تطبيق ويندوز عادي، وهي تحتوي على أسماء النوافذ المفتوحة داخل البرنامج، حيث يمكنك التنقّل بينها بضغط اسم أيٍّ منها.

كما أنها تحتوي على الأمر "إخفاء" Hide، الذي يقوم بإخفاء جميع مربعات الأدوات، ليخصص مساحة بيئة التطوير كلها لمحرر الكود أو مصمم النماذج.. وهذا الأمر لا يخفي مربعات النصوص نهائيا، بل يجعلها تنكمش إلى أشرطةٍ صغيرة على جانبي الشاشة، حيث يمكنك إظهارها بمجرد المرور بالفأرة على هذه الأشرطة.

**القائمة الرئيسية "استعلام (مساعدة)" Help Menu:**

وتحتوي على خيارات المساعدة المتنوعة ومنها:

**الأمر "المساعدة الفعّالة" Dynamic Help:**

يؤدّي لظهور نافذة المساعدة الفعّالة، وهي تُظهر لك تلقائيًا مواضيع الاستعلام التي تتعلق بالعملية التي تقوم بها حاليا.

**الأمر "فهرس" Index:**

يظهر لك فهرس مواضيع الاستعلام، حيث يمكنك كتابة عنوان أي موضوع، ليعرض لك شرحه.

**الأمر "محتويات" Contents:**

يعرض لك شجرة مواضيع الاستعلامات، لتختار منها ما تشاء.

**الأمر "بحث" Search:**

يعرض لك نافذة البحث، حيث يمكنك أن تكتب أي كلمة، ليتم البحث عنها في عنوان أو شرح أي موضوع.

**أهمّ نوافذ بيئة التطوير**

* + [**صندوق الأدوات Toolbox Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\1-%20صندوق%20الأدوات.htm)
  + [**متصفح المشاريع Solution Explorer:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\2-%20متصفح%20المشاريع.htm)
  + [**نافذة الخصائص Properties Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\3-%20نافذة%20الخصائص.htm)
  + [**نافذة المخرجات Output Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\4-%20نافذة%20المخرجات.htm)
  + [**نافذة الأوامر Command Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\5-%20نافذة%20الأوامر.htm)
  + [**نافذة المهام Task List Window:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\2-%20أهمّ%20نوافذ%20بيئة%20التطوير\6-%20نافذة%20المهام.htm)

**متصفح المشاريع The Solution Explorer:**

الترجمة الحرفية لكلمة Solution هي محلول.. والمقصود بها خليط المشاريع والمكونات التي يمكن أن تفتحها كلها في بيئة التطوير معًا في وقت واحد، ونظرًا لأنّه من السخيف أن نترجم هذا المصطلح بمتصفح المحاليل، فقد ترجمناه كما رأيت، على شرط أن يكون مفهومه واضحًا لنا.

ويعرض متصفح المشاريع كل عناصر برنامجك من مشاريع ومكونات، وما يندرج تحت كل منها من أعضاء وعناصر، بحيث يمكنك بنقر أيٍّ منها مرتين أن تعرضه في مصمم النماذج ـ لو كان من العناصر التي تعرض واجهة ـ أو فتحه في محرر الكود لو كان ملفًا نصيًّا فحسب.

كما يمكنك أن تُظهر القائمة الموضعية بضغط زر الفأرة الأيمن فوق أي عنصر، حيث يمكنك اختيار الأمر "خصائص" Properties، لتعرض نافذة الخصائص، لتعدّل خصائص هذا العنصر لو أردت.

ولو كان المشروع يحتوي على أكثر من نموذج، وتريد أن تجعل أحدها هو أول ما يظهر لك عند تشغيل البرنامج، فاضغط الأمر "اجعله كائن بدء التشغيل" Set As StartUp Object من القائمة الموضعية.

ولو كنت تتعامل مع أكثر من مشروع، وتريد أن تجعل أحدها هو المشروع الذي يتم تشغيله عند ضغط F5، فاضغط الأمر "اجعله مشروع بدء التشغيل" Set As StartUp Project من القائمة الموضعية.

كما يمكنك أن تضيف عناصر لأي مشروع بضغط الأمر "أضف عنصرا" Add Item من القائمة الموضعية، أو أن تحذف عنصرًا باستخدام الأمر "أخرجه من المشروع" Exclude From Project، وهذا الأمر لا يحذف ملف العنصر من على الجهاز، حيث يمكنك إعادته للمشروع من جديد في أي لحظة.. ولكي تحذفه نهائيًا من المشروع والقرص الصلب استخدم الأمر "حذف" Remove.

**صندوق الأدوات Toolbox Window:**

كما عرفت سابقا، فإن صندوق الأدوات يحتوي على جميع الأدوات التي تستخدمها لبناء واجهة التطبيق Application interface.

وعادةً ما يكون صندوق الأدوات متقلصًا في جانب الشاشة الأيسر، ولكي تعرضه، يجب عليك أن تمر بالفأرة على الشريط الصغير في هذا الجانب.

ويحتوي صندوق الأدوات على الأشرطة التالية:

**1تقارير جاهزة Crystal Reports:**

وبها الأدوات اللازمة لإنشاء تقارير جاهزة بمواصفات مختلفة، عن قواعد البيانات.

**"قواعد البيانات" Data:**

وتحتوي على الأدوات التي تستخدمُ للاتصال بقواعد البيانات وقراءة وتحديث بياناتها.

**"نماذج الويندوز" Windows Forms:**

وتحتوي على الأدوات التي تستخدم مع نماذج الويندوز.

**"نماذج الإنترنت" Web Forms:**

وتحتوي على الأدوات التي تستخدم مع نماذج الإنترنت.

**"أدوات HTML:**

وتحتوي على الأدوات التي تستخدم مع نماذج الإنترنت.. (وسنتعرّف فيما بعد على الاختلافات بينها وبين أدوات الإنترنت.

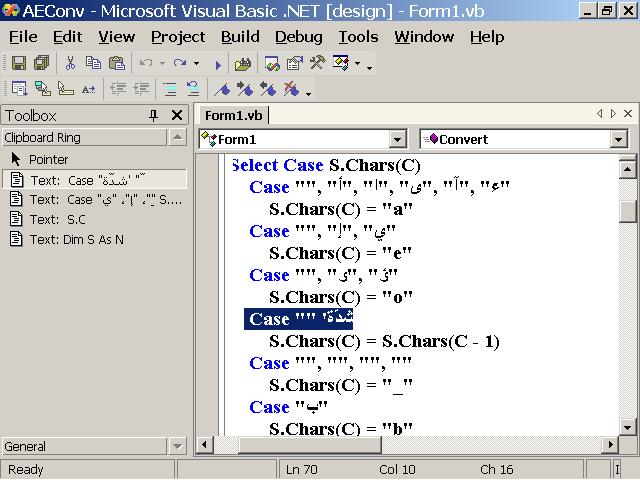
**"مخطَّط XML" XML Schema:**

**"محرر مربعات الحوار" Dialog Editor:**

**"مكونات" :Components**

**"حلقة مخزن القصاصات" Clipboard Ring:**

تحت هذا الشريط يتمّ الاحتفاظ بكلّ سطور الكود التي تنسخها من أيّ موضع.. إنّ هذا يمنحك ميزات أكبر من مخزن قصاصات الويندوز العادي Clipboard، حيث إنّ مخزن قصاصات الويندوز لا يحتفظ إلا بآخر عمليّة نسخ أو قصّ أجريتها، ولو نسخت أيّ كود جديد فسيتمّ وضعه في مخزن القصاصات بدلا من الكود السابق.. أمّا في حلقة مخزن القصاصات، فإنّ كلّ عمليّة نسخ يتمّ وضعها باسم على هذا الشريط، فإذا ما أردت نسخ أيّ نصّ منها، فما عليك إلا أن تقف بمؤشّر الكتابة في الموضع الذي تريد النسخ إليه في نافذة محرّر الكود، ثمّ تنقر مرّتين على العنصر المطلوب في حلقة مخزن القصاصات.



**"عامّ" General:**

هذا الشريط يكون فارغا مبدئيّا.. ويرجع لك الأمر، حيث يمكنك أن تضيف إليه الأدوات التي تستخدمها بصورة متكرّرة، أو الأدوات التي تنشئها أنت.

**نافذة الخصائص The Properties Window:**

تعرض كل خصائص العنصر المحدد حاليا، والقيم المناظرة لهذه الخصائص.

وأكثر الخصائص تقبل قيمة واحدةً فقط، رقميةً أو نصّيّة.. وإذا كانت القيم تنتمي لطائفة محددة من القيم لا تخرج عنها، يتم عرضها لك في قائمة منسدلة Drop-down List، حيث يمكنك الاختيار منها.

وهناك العديد من الخصائص الأخرى التي تحتاجُ لواجهةٍ خاصةٍ لاختيار مواصفاتها، مثل خاصية "اللون" Color، التي تعرض لك مربع حوار الألوان لاختيار اللون المناسب منه.

وهناك خصائص تعتبرُ "تجمّعا" Collection، بمعنى أنها تحتوي على مجموعة فرعية من الخصائص هي التي تحدد مواصفات الخاصيّة الأم.. ويمكن وضع هذه القيم بأكثر من طريقة، مثل خاصية "الخط"Font ، التي يمكن تحديد قيمها عبر "مربع حوار الخط" كما هو معهود، أو عن طريق ضغط علامة الـ "+" على يسار اسم الخاصية، حيث تتمدد الخاصية لتعرض أسماء عناصر الخط المختلفة، كالحجم والسمك وغيرهما، وبهذا يمكنك وضع قيم هذه العناصر كل على حدة.

ويمكنك إغلاق نافذة الخصائص في أي لحظة.. ولإعادة فتحها، اضغط الأمر "نافذة الخصائص" Properties Window من القائمة الرئيسية "عرض" View، أو اضغط زر الفأرة الأيمن على أي أداة أو كائن لتظهر لك القائمة الموضعية، ومنها اضغط الأمر "خصائص" Properties، أو بمنتهى البساطة اضغط F4 من لوحة المفاتيح.

يحدث أحيانًا أن تختفي أداةٌ كليةً تحت أداةٍ أخرى، بحيث لن يمكنك الوصول إليها بالفأرة لتحديدها ـ بضغطة واحدة ـ لاختيار خصائصها.. في هذه الحالة يمكنك استخدام القائمة المركّبة ComboBox الموجودة في أعلى نافذة الخصائص، فهي تعرض اسم النموذج، وأسماء كل ما عليه من أدوات.. اختر ما تريد منها، وغير ما شئت من خصائصه.

**نافذة المهام Task List Window:**

يمكنك فتح هذه النافذة بضغط الأمر Task List من القائمة الموضعيّة "نوافذ أخرى" Other Windows من قائمة الرئيسيّة "عرض" View.

وتظهر بهذه القائمة رسائل الخطإ التي تنتج عند ترجمة البرنامج.. ويمكنك أن تنقر مرتين على أي رسالة منها، لتذهب مباشرةً إلى سطر الكود الذي سبّبها، حتى تحاول إصلاحه.

ويمكنك كذلك أن تضيف مهامّك لهذه النافذة.. اضغط السطر الفارغ في بداية قائمة المهام، واكتب ما تريد: تعليقات وأشياء تحب أن تتذكرها، أو حتى مواقع إنترنت تهتم بها.

ورسائل الأخطاء الناتجة عن ترجمة الكود تختفي تلقائيًّا بمجرد تصحيح أسبابها، أما المهام التي أضفتَها، فعلى عاتقك أنت تقع مسئولية حذفها.

ويمكنك أن تضيف لهذا النافذة المهامّ التي تريد تذكّرها، عن طريق كتابة تعليقات Comments في المواضع المختلفة من الكود، على الصيغة التالية:

' TODO: أي تعليق يناسبك

بعد هذا ستجد تعليقك مذكورا في نافذة المهامّ، وعند الضغط عليه، ستنتقل مباشرة لموضع التعليق.

ولكي تتأكّد من عرض هذه المهمّة، اضغط على أيّ موضع من نافذة المهامّ بزرّ الفأرة الأيمن، ومن النافذة الموضعيّة اضغط القائمة الفرعيّة Show Tasks ومنها اضغط الأمر ALL.

ويمكنك أن تنشئ عناوين أخرى للتعليقات بخلاف TODO، وذلك بفتح القائمة Tools وضغط الأمر Options.. ستظهر لك نافذة الخيارات.. اختر العنوان Environment ومنه اختر العنصر Task List.. ستجد مربّع نصّ يمكنك أن تكتب بع عنوان التعليق الذي تريده، وزرّا يحمل كلمة Add عند ضغطه يتمّ إضافة هذا التعليق.

**نافذة الأوامر The Command Window:**

وقد ذكرنا سابقًا، أنك تستطيع أن تكتب بها أي سطر من الكود، وبمجرد ضغط زر الإدخال Enter للانتقال إلى سطر جديد، يتم تنفيذ هذا السطر.. وفي معظم الحالات، تستخدم هذه النافذة في وقت مقاطعة التنفيذ (وقت التوقف المؤقت) Break time، وهي حالة يصل إليها البرنامج بإيقافه مؤقتا عن التنفيذ بإحدى هذه الوسائل:

- إدراج نقطة توقف على سطر من الكود، وذلك بضغط F9 أثناء وجود مؤشر الكتابة في هذا السطر، حيث ستظهر دائرة حمراء في هامش النافذة بمحاذاة هذا السطر.. وعند تنفيذ البرنامج، سيتوقف التنفيذ مؤقّتا عند هذا السطر، حيث يمكنك اختبار البرنامج، ومعرفة قيم المتغيرات بالمرور فوقها بالفأرة، وما إلى ذلك.

- بضغط زري Ctrl+Puase من لوحة المفاتيح، لمقاطعة تنفيذ البرامج الطويلة.

- بمطالبة البرنامج بالتوقف عند تساوي أحد المتغيرات بقيمة معينة.

وسنتعرف على كل هذا بالتفصيل فيما بعد.. ما يهمنا هنا أن أي أمر تكتبه في نافذة الأوامر في وقت التوقف المؤقت يُعامل كأيِّ سطر من البرنامج، ويمكنك استخدام المتغيرات بنفس قيمها في البرنامج، وعند تغيير قيمتها في نافذة الأوامر تتغير قيمتها كذلك في البرنامج الأصليّ.

**نافذة المخرجات The Output Window:**

وتظهر بها المعلومات التي ترسلها أجزاء بيئة التطوير ـ مثل مترجم الكود Compiler ـ لتخبرك بحالة الوظيفة التي تؤديها.

وستظهر لك هذه النافذة في كل مرة تشغل فيها البرنامج Run the application.

ولو أحببت أن تعرضها بنفسك، فاضغط الأمر "المخرجات" Output من القائمة الفرعية "نوافذ أخرى" Other Windows تحت القائمة الرئيسية "عرض" View.

ويمكنك أن ترسل مخرجات لهذه النافذة من خلال الكود، باستخدام الأمر Console.WriteLine.. وتستخدم هذه الطريقة لمتابعة قيم المتغيرات داخل جزء به خطأ من البرنامج، بكتابة هذه القيم في نافذة المخرجات، ودراسة مدى صحتها وتماشيها مع مخطط برنامجك.

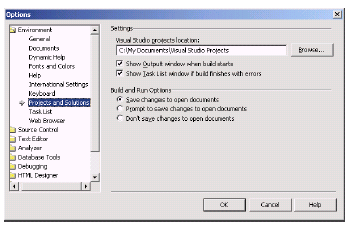
وسنرى كيفية استخدام هذا الأمر فيما يلي من فصول.

**خيارات بيئة التطوير Environment Options:**

إن بيئة التطوير الخاصة بـ VS.NET هي في غاية المرونة، ويمكن تغيير معظم خصائصها الشكلية والوظيفية لتناسب المستخدم.

تعالَ نرَ كيف يمكنك تغيير السمات الافتراضية لبيئة التطوير:

افتح القائمة الرئيسية "أدوات" Tools، واضغط الأمر الأخير فيها "خيارات" Options.. سيظهر لك مربع حوار الخيارات، حيث يمكنك أن تتعامل مع كل خيارات بيئة التطوير، مثل خطوط العناصر المختلفة (محرر النصوص، مربعات الحوار، أشرطة الأدوات... إلخ).



وتوجد مجموعة من أزرار التحويل Radio Buttons في أسفل مربع الحوار، وعن طريقها تستطيع اختيار متى يتم حفظ التغيرات التي تجريها في مشروعك.. مبدئيًّا، يتم حفظ التغييرات كلما شغلت البرنامج، ولكن لو اخترت الزر الأخير، فلن يتم حفظ التغييرات آليا، وعليك أن تحفظها أنت يدويا من وقت لآخر، بضغط الأمر "حفظ الكل" Save All من القائمة الرئيسية "ملف" File.

وعيك أن تستطلع الأشرطة الموجودة في مربع حوار الخيارات، فهي مباشرة وواضحة.

انتهينا الآن من استطلاع مكونات بيئة التطوير.. سننتقل الآن للأدوات Controls لاستطلاع أهم أعضائها Members.

**تعدد الأشكال Polymorphism**

* [**معناه:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\5-%20تعدد%20الأشكال\0-%20معناه.htm)
* [**قليل من الخصائص المشتركة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\5-%20تعدد%20الأشكال\1-%20قليل%20من%20الخصائص%20المشتركة.htm)
* [**قليل من الوسائل المشتركة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\5-%20تعدد%20الأشكال\2-%20قليل%20من%20الوسائل%20المشتركة.htm)
* [**قليل من الأحداث المشتركة:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\5-%20تعدد%20الأشكال\3-%20قليل%20من%20الأحداث%20المشتركة.htm)

**معناه:**

من السمات الرائعة في البرمجة بالكائنات Objects، الإمكانية المسماة بـ "تعدد الأشكال" Polymorphism.. وهذه الإمكانية تعني أنّه بينما يكون لديك العديد من الكائنات المختلفة، ستكتشف أن معظم خصائصها Properties ووسائلها Methods وأحداثها Events مشتركة في الأسماء ومتشابهة في الوظيفة مع بعض الاختلافات الطفيفة.

فمثلا ستجد أن النموذج ومربع النص والقائمة المركّبة، لكل منها خاصية "نص" Text، يمكنك أن تكتب النص فيها الذي تريدها أن تعرضه.. ولكنك ستلاحظ أن النموذج يعرض النص في شريط بأعلاه، بينما يعرضه مربع النص في داخله، بينما تعرضه القائمة المركبة في مربع النص الخاص بها.. إذن فهناك اختلافات في وظيفة هذه الخاصيّة في كل من هذه الكائنات.. الجميل أنك معزول عن هذه الاختلافات، وليس مطلوبا منك أي شيءٍ للتعامل معها.

هذه ميزة كبيرة، فهي توفّر عليك وقتا طويلا للتعلم، فبمجرد أن تعرف وظيفة خاصية Text في مربع النص مثلا، ستكتشف أنّك قد تعلمت وظيفتها في أي أداةٍ أخرى لها نفس الخاصيّة، مثل "اللافتة" Label وزر الاختيار CheckBox وزر التحويل Radio Button... إلخ.

يفيد هذا أيضا في سهولة حفظ أسماء الخصائص ووظائفها، فلو كان لنفس الوظيفة اسم مختلف في كل أداة لكانت كارثة!

**قليل من الخصائص المشتركة A Few Common Properties:**

**خاصية "الاسم" Name:**

وهذه الخاصية تظهر في أعلى نافذة الخصائص دون مراعاة للترتيب الأبجدي، وذلك لأهميتها.

وهي عبارة عن الاسم البرمجيّ للأداة، وهو الذي تكتبه في الكود.. مثال:

لو أسميت مربع نص Text1، يمكنك أن تغير النص المكتوب فيه برمجيا بجملة كالتالية:

**Text1.Text = "أي شيء"**

أيّ أن هذه الخاصية هي مفتاحك لاستخدام باقي خصائص ووسائل الأداة.

**ملاحظة هامة:**

نظرًا لأن اسم الأداة سيظهر كثيرًا في البرنامج، يجب عليك اختيار اسم يكون موحيا، حتى يمكنك فهم البرنامج عند قراءته فيما بعد.. والطريقة المتفق عليها في التسمية هي تكوين الاسم من مقطعين:

**1- المقطع الأول:** بادئة تدل على نوع الكائن، عبارة عن حرفين أو ثلاثة لاختصار اسمه.. وسنعطيك بعض الاقتراحات في الجدول التالي:

|  |  |
| --- | --- |
| Button | Bt |
| CheckBox | Chk |
| ComboBox | Cmb |
| Form | Frm |
| Label | Lb |
| ListBox | Lst |
| Menu | Mnu |
| RadioButton | Rdo |
| RichTextBox | Rt |
| TextBox | Txt |

وهكذا...

**2- المقطع الثاني:** كلمة تدل على وظيفة الأداة، حتى يمكن فهمها بمجرد قراءة الاسم.. أمثلة:

- لو لديك مربع نص يكتب فيه المستخدم اسمه، فسمِّه TxtUserName أو TxtName لو أردت الاختصار.

- ولو لديك لافتة تعرض مجموع عددين، فسمّها LbSum.

- ولو لديك قائمة مركبة تعرض أسماء الدول العربية، فسمّها CmbArabCountries.

- ولو لديك زر مكتوب عليه "موافق"، فسمّه BtOk.

وهكذا.... .

**خاصيّة "من اليمين لليسار" RightToLeft:**

وهي من أهمّ الخصائص لنا كمبرمجين عرب، حيثُ إنّها تسمحُ لك باستخدام الأداة في التطبيقاتِ العربية، فتكون محاذاتها من اليمين لليسار، ويمكنك كتابةُ النصوص العربية بها بدون مشاكل.

**خاصيّة "الخط" Font Object:**

وهي تحدد المواصفات التي سيعرض بها الخط في الأداة، في وقتيِ التصميم والتنفيذ، كاسم الخطّ وحجمه وهل هو سميك أو مائل... إلخ.

**خاصية "لون الخلفية" BackColor:**

تحددُ لونَ خلفيّة الأداة.. وقد رأينا في بدايةِ هذا الفصلِ كيفيّة اختيار الألوان من نافذة الخصائص.. ويمكن كذلك تغيير هذه الخاصيّة من الكود.. والجملة التالية تغيّر لون مربّع النصّ إلى الأحمر:

**TextBox1.BackColor = Color.Red**

ويمكن جعل خلفيّة الأداة شفّافة Transparent، إذا كانتِ الأداة تقبلُ هذا، مثل اللافتة Label ومربّع الاختيار CheckBox.. ولفعل ذلك من نافذة الخصائص، حدّد خاصيّة BackColor واضغط زرّ الانتقال المجاور لها، وفي مربّع اختيار اللون اضغط شريط Web.. ستجد أنّ أول لون في قائمة ألوان هذا الشريط هو اللون الشفّاف Transparent.. اضغط هذا اللون بالفارة.

كما يمكنك فعل ذلك من الكود، كالتالي:

**Label1.BackColor = Color.Transparent**

إنّ جعل خلفيّة الأداة شفّافة يجعلها تكتسب لون النموذج.. ولو وضعت على النموذج صورة، فلن تحجبها الأداة في هذه الحالة، ممّا سيمنح النموذج شكلا أجمل.

**خاصية "لون الخطّ" ForeColor:**

تحدّد هذه الخاصيّة لون النصّ المكتوب في الأداة.. ويمكن تغييرها بطريقة مشابهة لتلك التي يتمّ بها تغيير لون خلفيّة الأداة، سواء من نافذة الخصائص، أو من الكود.

**خاصية "فعّالة" Enabled:**

إن الأدوات فعّالة في الوضع التلقائيّ، بمعنى أنّها تستجيب للمستخدم عن طريق الفأرة أو لوحة المفاتيح.. ولكن هناك حالات تحتاج فيها لمنع المستخدم من استخدام أداة معينة، حتى لا يتسبب في خطإٍ ما.. فمثلا، لو لديك نافذة تطلب من المستخدم كتابة اسمه وكلمة المرور الخاصة به، وبها زران: "موافق" و"إلغاء"، فمن المنطقيّ أن تمنع المستخدم من ضغط زر "موافق" إذا لم يكتب الاسم أو كلمة المرور.



هنا يمكنك جعل الزر غير فعّال، بكتابة أمر على شاكلة:

**BtOk.Enabled = False**

وبهذا يظهر الزر في لون رمادي، ويمتنع عن الاستجابة للمستخدم.

**خاصيّة "الحجم" Size:**

لقراءة أو تغيير حجم الأداة.. وهي عبارة عن كائن يحتوي على خاصيتين: "العرض" Width و"الارتفاع"Height .

**الخاصية الإضافية Tag:**

وهي مخصصة لتخزين أي نوع من المعلومات تريد وضعه داخل الأداة.

**خاصية "نص" Text:**

تقرأ أو تغيّر النص الذي يظهر على الأداة.

**خاصية "التوقف بضغط زر Tab" TabStop:**

أداة واحدة فقط يمكنها أن تكون نشيطة Active في لحظة معينة، بمعنى أنها تستقبل ما يتم ضغطه من الأزرار من لوحة المفاتيح أو الفأرة.. وللانتقال من أداة لأخرى يمكنك أن تضغط الأداة بالفأرة لتنشيطها.. ولكنّ كثيرًا من المستخدمينَ يفضّلونَ استخدام لوحة المفاتيح، لأنّها تكونُ أسرعَ خاصةً عندَ إدخال كمّ كبير من البيانات.. لهذا فإنّ بإمكانهم الانتقال بينَ الأدوات باستخدام لوحة المفاتيح، وذلك بضغط زر Tab (للتحرك للأداة التالية) أو زري Shift+Tab (للتحرك للأداة السابقة).

هنا يمكنك أن تسمح باستثناء الأداة من هذا التصرف، بجعل قيمة خاصية TabStop خطأ False.. في هذه الحالة لن يستطيع المستخدم الوصولَ إلى الأداة إلا باستخدام الفأرة.

**خاصية "ترتيب الانتقال" TabIndex:**

وهي قيمة رقمية تستخدم لتوضيح موقع الأداة في "ترتيب الوصول بضغط زر Tab" Tab Order، وهو يبدأ من الصفر.

إن الأداة ذات الرقم الأصغر، هي تلك التي ستستقبل العلامة الضوئية عند بدء ظهور النموذج.

ولو ضغطت زر Tab مرة واحدة، فستنتقل العلامة الضوئية للأداة التي تمتلك أقل رقم يكبر عن الرقم الحالي.

**خاصية "مرئية" Visible:**

يمكن استخدام هذه الخاصية لعرض أو إخفاء الأداة، بوضع قيمة هذه الخاصية بصواب True أو خطأ False على التناظر.

**قليل من الوسائل المشتركة A Few Common Methods:**

واضح من اسمها أنها "وسيلة" لأداء وظيفة معينة.. والوسائل هو الاسم الذي يندرج تحته:

- الإجراء Procedure، وهو وظيفة يتم تنفيذها دون أن تستقبل منها قيمة عائدة Return Value.

- والدالة Function، وهي مثل الإجراء، وإن كنت تستقبل منها قيمة عائدة Return Value.

ولا تحتوي أدوات الويندوز Windows Controls على الكثير من الوسائل، بينما الكائنات Objects تمنح وسائل أكثر ـ كما سنرى فيما بعد.

إن كل شيءٍ ـ حتى المتغيرات بل والأرقام ـ هي كائنات، ولكل منها مجموعة خاصة من الوسائل، يمكن استخدامها بكتابة نقطة "." بعد اسم الكائن، ثم كتابة اسم الوسيلة.. وكما ذكرنا مسبقا، فإن كتابة نقطة بعد اسم الكائن، سيؤدي لظهور قائمة بكل خصائصه ووسائله.

فمثلا يمدنا المتغير النصيّ String variable بهذه الوسائل:

- "طول النص" Length:

وهي دالة لحساب طول النص.

- "حوّل للحروف الكبيرة" ToUpper:

وهي دالة تكّّون نصًا جديدا كل حروفه كبيرة Capital.

- "حوّل للحروف الصغيرة" ToLower:

وهي دالة تكّّون نصًا جديدا كل حروفه صغيرة Small.

ولترى كيف تعمل هذه الدوال، أنشئ تطبيقا جديدًا، وضع زرا على النموذج، واكتب الكود التالي في حدث ضغط الزر:

**Console.WriteLine("Visual Basic".Length)**

**Console.WriteLine("Visual Basic".ToUpper)**

**Console.WriteLine("Visual Basic".ToLower)**

اضغط F5 لتشغيل البرنامج، واضغط الزر.. سترى ما يلي في شاشة المخرجاتOutput window :

12

VISUAL BASIC

visual basic

**ملحوظة:**

إذا لم تكن شاشة المخرجات ظاهرة، فاضغط: عرض/نوافذ أخرى/نافذة المخرجات View/Other Windows/Output.

والآن، لنَرَ قليلا من الوسائل المشتركة بين معظم الأدوات:

**"تنشيط" Focus:**

كلمة Focus تعني بؤرة أو يركّز، ويقصد بها هنا أن تكون الأداة أو العنصر هي بؤرة اهتمام نظام التشغيل، بمعنى أن تكونَ نشيطة، تتلقى ضغطات الفأرة وأزرار لوحة المفاتيح.

واضح إذن أن هذه الوسيلة تنشّط الأداة التي تناديها، بغض النظر عن هويّة الأداة التي كانت نشيطةً من قبل.. مثال:

**TextBox1.Focus**

ويمكن اصطياد العلامة الضوئية في أداة معينة، إلى أن يكمل المستخدم إدخال قيمة ما، وذلك عن طريق إعادتها للأداة مرة أخرى في حدث "الخروج" Leave.. خذ هذا المثال، الذي لا يسمح للمستخدم بمغادرة مربع النص، إلا إذا كان يحتوي على أي قيمة رقمية مقبولة:

**Private Sub TextBox1\_Leave(ByVal sender As Object, \_**

**ByVal e As System.EventArgs) Handles TextBox1.Leave**

**If Not IsNumeric(TextBox1.Text) Then TextBox1.Focus()**

**End Sub**

والدالة "إنه رقميّ" IsNumeric()، تعطي القيمة صواب True إذا كان معاملها ـ المرسل بين قوسيها ـ هو قيمة رقمية مثل 45 و 344.44، فإن لم يكن، فهي تعطي القيمة خطأ False.

**"مسح المحتويات" Clear:**

وهي تمسح كل محتويات الأداة لتفرغها تمامًا.. فمثلا تقوم هذه الوسيلة بمسح محتويات مربع النص بجعل خاصية "نص" Text فارغة.

**"إخفاء" و"عرض" Hide/Show:**

وواضح من اسميهما وظيفتاهما.. وهما تقومان بتغيير قيمة الخاصية "مرئية" Visible، فتجعلها الوسيلة الأولي False، وتجعلها الثانية True.

**"اضغط الأداة" PerformClick:**

إذا أردت أن تضغط زرا من الكود، فاستدع هذه الخاصية من الأداة.

**"تحجيم" Scale:**

يمكن استخدام هذه الوسيلة لتكبير أو تصغير الأداة بنسبة معينة.

مثال: هذا الأمر يصغّر مربع النص إلى 75% من حجمه الحاليّ:

**TextBox1.Scale(0.75)**

**قليل من الأحداث المشتركة A Few Common Events:**

إن أوامر برنامجك، تُكتب أساسا في مجيبات الأحداث Event Handlers، حتى يمكن للبرنامج أن يتفاعل مع المستخدم.

ورغم أن لكل أداةٍ عددا هائلا من الأحداث، فإنك نادرا ما تبرمج أكثر من حدث واحد منها، بل إن كثيرا من الأدوات لا تحتاج لكتابة أي أوامر لها.

وهذه أكثر الأحداث استخدامًا:

**حدث "الضغط" Click:**

**حدث النقر المزدوج DoubleClick:**

**حدث "الدخول" Enter:**

يحدث عندما تستقبل الأداة العلامة الضوئية وتصبح نشيطة.

**حدث "الخروج" Leave:**

يحدث عندما تغادر العلامة الضوئية الأداة، وتصبح غير نشيطة.

وغالبا ما تكتب هنا الكود الذي يتحقق من صحة محتويات الأداة قبل مغادرتها.

**حدث "دخول الفأرة" MouseEnter:**

يحدث عندما يعبر مؤشر الفأرة المساحة التي تشغلها الأداة، وهو يطلق مرة واحدة فور دخول مؤشر الفأرة، ولا يطلق مرة أخرى حتى يغادرها ثم يدخلها من جديد.

ولكن لو أردت مراقبة حركة الفأرة فوق الأداة، فعليك باستخدام الحدث "حركة الفأرة"MouseMove .

**حدث "مغادرة الفأرة" MouseLeave:**

ويحدث عندما يخرج مؤشر الفأرة عن حدود الأداة.

**الحدث "تغيّر...." XXXChanged:**

حيث البادئة XXX هي مجرد رمز عام، يمكن استبداله باسم خاصية ما، مثل TextChanged و BackColorChangedو FontChanged... إلخ.

جليٌّ أن هذه الأحداث تحدث عندما تتغير قيمة خاصيّةٍ ما سواء غيّرها المستخدم، أم غيرها المبرمج من الكود.

**أنواعُ المشاريع**

يمكنك أن ترى كلَّ أنواعِ المشاريعِ المتاحةِ ـ بخلافِ تطبيقات ويندوز ـ في مربع حوار "مشروع جديد"، وهي كالتالي:

**[مكتبة الخلايا Class Library:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\1- مكتبة الخلايا.htm)**

**[مكتبة أدوات تحكم ويندوز Windows control library:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\2- مكتبة أدوات تحكم ويندوز.htm)**

**[خدمة ويندوز Windows service:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\3- خدمة ويندوز.htm)**

[**تطبيق الاختبار Console application:**](file:///C:\Users\Administrator\Desktop\vb.net\B-%20أساسيّات%20اللغة\01-%20التعرّف%20على%20اللغة\6-%20أنواعُ%20المشاريع\4-%20تطبيق%20الاختبار.htm)

**[تطبيق الإنترنت ASP.NET Web Application:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\5- تطبيق الإنترنت.htm)**

**[خدمة الإنترنت ASP.NET Web Service:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\6- خدمة الإنترنت.htm)**

**[مكتبة أدوات تحكم الإنترنت Web Control library:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\7- مكتبة أدوات تحكم الإنترنت.htm)**

**[مشاريع أخرى:](C:\\Users\\Administrator\\Desktop\\vb.net\\B- أساسيّات اللغة\\01- التعرّف على اللغة\\6- أنواعُ المشاريع\\8- مشاريع أخرى.htm)**

**مكتبة الخلايا Class Library:**

وهي مكوّن أساسيّ لبناء الكود، ليس لها واجهة مرئيّة، ولكنّها تؤّدى وظائف معيّنة لبرنامجك.

وباختصار، فإنَّ الخليّة هي مجموعةٌ من الدوالّ والإجراءات التي يمكنُ إعادة استخدامها في أيّ مشروع، بحيثُ يوفّر المبرمجُ على نفسِه مشقّةَ إعادة كتابتها مرة أخرى.

**ملحوظة لمبرمجي VB6:**

إنّ مكتباتِ الخلايا مكافئةٌ لمشاريع ActiveX.DLL و ActiveX.EXE في VB6.

**مكتبة أدوات تحكم ويندوز Windows control library:**

إنّ أدة ويندوز ـ أو الأداة على سبيل الاختصار ـ هي عنصر أساسيٌّ في واجهة البرنامج، مثل مربع النص أو الزر.

فإذا ما وجدتَ أنَّ الأدواتِ التي يمنحُها لك VB.NET ـ وهي التي تظهر في مربع الأدوات تلقائّيا ـ لا تمنحُكَ الوظيفةَ التي تريدُها، ففي هذه الحالةِ يمكنُكَ أن تبنيَ الأدواتِ الخاصّةَ بك.. إنَّ هذا يبسّطُ عملَكَ فيما بعد، حيثُ لن تُضيعَ المزيدَ من الوقتِ والجهدِ في تصميمِ أجزاءٍ من واجهاتٍ تتكرّرُ كثيرًا.

بل يصلُ بك الأمرُ ـ لو كنتَ خبيرًا في تصميمِ الأدواتِ الخاصّة، إلى أنّك تستطيعُ بيعَها للشركاتِ والمبرمجينَ الآخرين.. إنَّ صفحاتِ مجلاتِ الكمبيوتر تزخرُ بالإعلاناتِ للعديدِ من الأدواتِ الخاصّةِ المتطوّرة، التي تُكملُ عملَ الأدواتِ الأصليّة.

**خدمة ويندوز Windows service:**

وهو الاسم الجديد لـ Services NT القديمة.. وهذا النوع من المشاريع، ما هو إلا برامج طويلة التشغيل تعمل في خلفية نظام التشغيل، دون أن تعرض واجهةً مرئية.. ويمكن لهذه الخدمات أن تعملَ تلقائيًّا بمجرد تشغيل الجهاز، كما يمكن إيقافها لفترةٍ ثم إعادة تشغيلها.

إن البرنامج الذي يتفاعل الملفات على الجهاز ويغيرها، هو مرشح رئيسيّ لأن يكون خدمة ويندوز، حيث يمكن لهذه الخدمة أن تؤدي بعض العمليات، فور أن ينقل المستخدم بعض الملفات إلى مجلد معين، كأن تنسخ الملفاتِ أو تقرأ محتوياتها، أو تحدّث محتويات قاعدة بيانات... وهكذا.

**تطبيق الاختبار Console application:**

وهو تطبيقٌ ذو واجهةِ استخدامٍ محدودة جدًا، هي أقربُ ما تكونُ لشاشةِ الدوس DOS السوداء، حيثُ يتمُّ فيها إخراجُ النواتجِ وإدخالُ البيانات، على سبيل اختبارها.

**تطبيق الإنترنت ASP.NET Web Application:**

إن تطيبق الإنترنت هو واحد من أكثر السمات الجديدة في VS.NET إثارةً، وهو عبارة عن برنامج يمكن تشغيله على شبكات الإنترنت، يمكن طلب خدمات معينة منه عن طريق متصفحات الشبكة.. وأوضح مثال لذلك، هو محل كتبٍ يعرض ويبيع على الإنترنت.

ويجب على التطبيق الذي يعمل على الإنترنت أن يستقبل طلبات العملاء، ويردّ عليها في صورة صفحة من صفحات الإنترنت HTML.

ومثل هذه التطبيقات ليست بجديدة، ولكن الجديد هو أن ASP.NET صار يعزل المبرمج عن كثيرٍ من تفاصيل بنائها، مانحًا الكثير من السهولة والسرعة، ومحولا العملية إلى ما يشبه كثيرا بناء تطبيقات ويندوز التقليدية.

وسنناقش بالتفصيل خدمات وتطبيقات الإنترنت في آخر جزء من هذا الكتاب إن شاء الله.

**خدمة الإنترنت ASP.NET Web Service:**

وهي لا تكافئ بحالٍ خدمات ويندوز، بل هي عبارةٌ عن برنامجٌ يعملُ على خوادم الإنترنت Internet Servers لاستقبال الطلبات، تمامًا مثل تطبيق الإنترنت، ولكن دونَ أن يرد بصفحةٍ من صفحات الإنترنت، بل يرسل ناتجَ العملية الحسابية، أو المعلومات المطلوبة من قاعدة بياناتٍ ما، وعادةً ما تُطلب هذه الخدمات من قبل خادم إنترنت آخر، حيث سيتولّى هو التعامل مع البيانات.

ولتوضيح الفارق نضربُ هذا المثال:

إن تطبيق الإنترنت الذي يتلقى منك استعلامًا عن كل كتب (توفيق الحكيم) سيعرض لك صفحةً عليها أسماء هذه الكتب.. أما خدمة الإنترنت التي تتلقى منك نفس هذا الاستعلام، فسترسل إليك ملف XML يحتوي على أسماء الكتب المطلوبة، حيثُ يمكن لتطبيقك في هذه الحالةِ أن يتعامل مع البيانات ويعرضها في صفحةٍ جديدةٍ بالشكل الذي تريده، ويرسلها للمستخدم الذي تريده، أو ربما تعرضها على نموذج ويندوز عاديّ.

**مكتبة أدوات تحكم الإنترنت Web Control library:**

كما أنك تستطيع أن تبني أدوات الويندوز الخاصة بك لتستخدمها على نماذج ويندوز، تستطيع أيضًا أن تبني أدوات الإنترنت الخاصة لتستخدمها على نماذج وصفحات الإنترنت.

**مشاريع أخرى:**

أما القوالبُ الثلاثة المتبقية في مربع حوار "مشروع جديد"، فهي:

1- مشروع فارغ Empty Project.

2- مشروع إنترنت فارغ Empty Web Project.

3- مشروع جديد في مجلد موجود New Project In Existing Folder.

وهي ليست أنواع مشاريع بالمعنى الفعليّ، بل مجرد طريقة لتنظم مشاريعك الجديدة بنفسك، فحين تختارُ واحدا منها، ينشئ VS.NET مجلدا جديدا يحمل اسم المشروع، وينشئ فيه بعض الملفات القليلة التي يحتاجها البرنامج من هذا النوع.. وعلى عكس الأنواع الأولى ـ حيث تقومُ VS.NET بإنشاء عناصر افتراضيّةٍ والملفاتِ اللازمة لها، كإنشاء نموذج مبدئي في تطبيق ويندوز، وإنشاء أداة المستخدم في تطبيقات أدوات ويندوز ـ تحمّلك هذه الأنواع الثلاثة مسئولية إنشاء وإضافة ما يلزمك من العناصر والملفات.