

Microsoft

**Visual
Studio 7.0**

Visual Basic .net

إستثمر اللغة بكل طاقتها

بقلم محمد عبد الناصر خطيب



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

((سُبْحَانَكَ لَا عِلْمَ لَنَا إِلَّا مَا عَلَّمْتَنَا
إِنَّكَ أَنْتَ الْعَلِيمُ الْحَكِيمُ))

الوجيز في الجديد

استثمر اللغة بكل طاقتها

الطبعة الأولى ٢٠٠٥

*حقوق كتاب "الوجيز في الجديد" محفوظة للمؤلف ولا يحق لأي شخص أو جهة رسمية إعادة نشر هذا الكتاب أو جزء منه بأي وسيلة دون الإذن الخطي من المؤلف

*أسماء البرامج المذكورة في هذا الكتاب هي علامات تجارية مسجلة لأصحابها والمؤلف يحترم هذه العلامات ويقر بملكيته سواء كانوا أفراد أو شركات أو أي جهة تنظيمية ولم يتم ذكرها للاختصار

*تم اختبار المادة العلمية في هذا الكتاب والتحقق منها ومراجعتها إلا أن المؤلف غير مسؤول بأي شكل من الأشكال عن الأضرار الناتجة عن سوء التطبيق أو الأكواد المستخدمة

*جميع الآراء الموجودة في هذا الكتاب تعبر عن رأي المؤلف الشخصي حتى ولو لم توثق بأمثلة أو أدلة.

*هذا الكتاب مجاني عسى الله ينفع به كل مسلم...

المحتويات

٥	إهداء
٦	شكر و تقدير

الفصل الأول

١١	إنشاء التوابع
١٨	دوال التحويل
٢٣	دوال موارد اللغة
٢٤	الدوال الرياضية
٣٢	الدوال الرياضية القديمة
٣٤	دوال الوقت
٤٣	دوال الموارد المالية
٥١	دوال النصوص (السلاسل النصية)
٦٢	دوال التعامل مع الملفات
٧٤	دوال المعلومات
٨٥	دوال التفاعل

الإهداء

أهدي هذا الكتاب والذي أرجو من الله أن يتقبله
إلى شعبة النور أمي التي مهما فعلت
بقيت الأفعال عاجزة عن رد الجميل.

مع تحياتي:



شكر و تقدير

الحمد لله الذي لا يحمد على مكروهه سواه.
وبداية أتوجه بالشكر لكل المناضلين لإيصال الحق إلى المسلمين ولطلبة العلم
والعالمين،
وأتوجه بالدعاء إلى الله أن يستفيد المبرمجين ويقدمهم خطوة على الأقل إلى الأمام.
ولن أنسى أن أشكر: الأخ: تركي العسيري الذي فجر عندي كل الأحداث التي تشد
المبرمج نحو الأمام
ولا أنسى الكثير من المبرمجين الذين أروني أن الحماس طريق نهايته جميلة وإلى
كل صديق

لمن هذا الكتاب

أولاً - إذا كنت من المبتدئين فسيكون لك مرجع تلتهم منه الأفكار. فقط عليك التركيز فالأمثلة المتبعة أبسط الأمثلة ؟

ثانياً: فإذا كنت من أصحاب الإصدارات القديمة ولم تجرأ على التقدم للأمام فلعله كتاب تطوير لنفسك Update؟

ثالثاً: أما إذا كنت من المتقدمين فلن يكون إلا أوراق إنعاش لك وراحة لِنفسي وإعطائها الجرأة على التقدم في كتب أخرى بإذن الله

أما إذا كنت من المنتقدين وأرجو من الله أن لا تكون. أن تنظر إلي نظرة خير بإذن الله؟

الأفكار التي ستعرض فيه

- ١- أفكار جديدة وربما أول مرة تسمع عنها؟
- ٢- حلول لحوالي العشرات من الأمور المستصعب حلها؟
- ٣- شرح خاص لعدة أمور ومنها تركيب التوابع وإنشاء الملفات وغيرها من الأمور التفاعلية؟
- ٤- أقول أن الدوال التي سوف أذكرها ليست كل الدوال فكما تعلم أصبحت مفهوم البرمجة معظمه دوال فلذلك قدمت شرح أصناف رئيسية قديمة كانت أم جديدة؟
- ٥- حاول التركيز على الأفكار فتطبيقها يكاد يكون أسهل مما تتوقع فلن تحتاج لقراءة الأمثلة أو تطبيقها أمر صعباً.
- ٦- إن هذا الكتاب هو جزء من سلسلة تحوي نفس العنوان الوجيه في الجديد.
- ٧- سيتم بإذن الله ذكر إصدار الذي تعمل فيه الدالة ووجودها من ٢٠٠٢-٢٠٠٥.

مقدمة

منذ أول يوم رأيت هذه اللغة تفجرت عندي المواهب المخبأة أيام النحت على الخشب وتجميع الدارات الإلكترونية البسيطة لتشكل عندي روى خاصة.
صحيح بأن مشواري البرمجي لم يمضي عليه سوى خمسة أعوام ثلاثة منها قضيتها في الإصدار القديم . ولكن هدفي هو إرضاء الله
صحيح أن للبرمجة أهداف كثيرة لطالما سعى الإنسان إليها ولكن ليس لكل شيء حل فيها مادامت من الجوهر مفيدة، ولكن منها تستطيع العمل والتطوير لتصل لحقيقة واحدة، أن العلم فائدة.
وأحببت أن أقدم شرحا لمعظم الأوامر التي ستغني مبرمجينا عن التفكير بحلها لتدفعهم نحو أشياء ذات أهمية وما تبقى دون ذلك فهو خاص برؤى تنير الطريق نحو مستقبل واعد وأقدم كلمات حب لهذه اللغة وهي وجيز من كلمات كتبتها

((تذكرتك وذكراك عطرة ورسمتك جميلة لا بشعة كلام كتبتنه والعقل فيك مشتاق في كل زاوية فكرة صحيح أنك كثيرة الأخطاء ولكن تبقى ذكراك أجمل صحيح أنك كثيرة الأخطاء ولكن الحياة معكي أفضل بدونك أنسى وأحلم باليوم الذي سوف تصبحين أجمل فأجمل تقل مع السنين أخطاءك ويكبر عقلك وتتسع ذاكراتك لتسيطر على الجهاز المكسل وتصبحين أقوى وأذكى والتعامل معك يصبح أسهل لاداعي للعصبية الحمقاء أحيانا فرسالة الطوفان أكاد أكرها وأخرى وقت التشغيل تكاد لاتفهم وبعد عامين على اللقاءيا أصبحت أسمن وبالحجم إزدتي حتى طررت لشراء هارد أكبر ولكن سيأتي يوما لنحقق الحلم ونصبح أفضل.....))

مدخل إلى الفيجوال بيسك ٧ (net)

لن أبدأ حديثي عن قدرات الإصدار الجديد وأسترسل لها بالمدح ولكن يمكن أن نعتبره كما قال الكثير ومنهم الأخ تركي العسيري أنها لغة جديدة ولكن بScript قديمة وأزيد على ذلك وأسميه خط تطوير تجاري فالغرض منها تغيير الواقع البرمجي من منحى بسيط إلى آخر معقد لن تستطيع الأفراد بمفرده العمل عليه بسهولة فسوف يطيل والملل سوف يصير أقرب إلى نفس المستخدم ولكن كما تعلم أضافت ما يسمى السي شارب فهل تعرف لما: أولاً أنه معظم الذين كانوا من مبرمجين الإصدارات القديمة هم الذين إستمرو في إصدار الفيجوال نت وأما جل المبتدئين فإما انخرطوا في سياق اللغات المشهورة عالمية والجديد منها

فلو لاحظت الدعم التي تقدمه مايكروسوفت في لمبرمجي الفيجوال سي هي أكبر بكثير من الدعم الذي تقدمه لمبرمجي الفيجوال بيسك وشاهدنا ذلك في الإصدار القديم لا أقصد ال Help ولكن أقصد الأدوات وبرامج إضافية ومكتبات تخزين .

وكما قلت لك فإنه مجرد الحفاظ على مبرمج الفيجوال بيسك ولكن على ما أظن أن الهاكرز والكراكز لن يستطيعوا برمجة أدواتهم على هذا الإصدار إلى أن يتم إنزال نظام حاوي على ملفات الFramework وأدوات إضافية تسمح بتشغيل برامجنا الجميلة أظنها إلى بعد ٢٠٠٦ أي WindowsLonHorn إن شاء الله.

يعني أنا برمجتنا ستبقى نزيهة إلا ذلك التاريخ أو بعده مع العلم أن بعضاً ربما أو أكثر من ذلك يحبون العمل على Windwos98 و Win Me.

لا أحبب الهمم فهذا الطريق الشائك لمبرمجين البيسك هم وحدهم القادرين على اجتيازه. ولكن ماذا أقدر أن أفعل في هذا الإصدار الجديد بما أنه متضمن برامج تخص تطبيقات الإنترنت فسوف تستطيع صناعة البرامج من البرامج الخدمية وقواعد البيانات مرورا ببرامج خدمات للنظام وصفحات الإنترنت النشطة ونهاية بتطبيقات الموبايل و ASP. سأقول لك أمران أن هذا الكتاب بداية لك وتحدي للغزاة الطامعين في إوقاعنا بعيداً عن تعليمات البيسك التي أحبها

هل سأشرح لغة البرمجة كلها أم ماذا؟

بداية:

هذا فصل من اللغة فلن نعيد كتابة ما كتب من الكتب ولكن أخبرنا في عنوان الكتاب أنه وجيز للجديد في مجال التوابع أي الدوال ولذلك علينا أيضا دعم كل ما يتعلق بها .. ومن هنا عليك أن تكون متقن لما يلي حتى تفهم الكتاب حق الفهم؟

١- أنواع المتغيرات (عددية) (Short-Long-Intreger-Double-Single-Int32-Int64) أو نصية (String- Char) أو تاريخ (Date) أو كائنيه (Object)

٢- الفرق بين تعليمات التصريح (Private ,Dim, Public , Friend ,Protected , Declare)س

٣- تشكيل المصفوفات البدائية ذات البعد الواحد وكيفية عملها لأن الخوض في حماها يكون أمر شاق ولا نبغي الإطالة وإذا كنت من الذين لا يريدون أن يعرفوا شيء فيها. سأجيبك بلغة (بدائية) شرح عندما ترى بجانب المتغير أثناء تعريفه قوسين فاعلم أنه مصفوفة وإذا وجدت في داخله عدد فإنه يمثل عددها وإذا وجدت بداخله عددين يفصل بينهما فاصل فإنهما ينتميان إلى مصفوفة ثنائية وهكذا دون توقف إلى ٦٥ وأكثر من ذلك وسوف يمر معنا مصفوفة ببعدين فقط في هذا الكتاب في دالة وحيدة في دوال التفاعل

Dim F ()

Dim T (,)

Dim H (, ,)

٤- الفرق بين أماكن التصريح وكيفية إحضار ذلك التابع سواء من Module أو من Class أو غير ذلك.

٥- تكوين التراكيب من نوعين Enum و Structure

وإذا لم تكن قد سمعت بهم أو سمعت ولكن لم تعرف يهما أما الأولى فلأهميتها شأن.. ربما تقول ما فائدتها في تشكيل التابع سأجيبك عليها بعد التعرف على شكل تشكيل التابع

الدوال أو التوابع البرمجية

(الدوال أو التوابع) وهي قصاصات برمجية يتم ضمنها تعريف متغيرات وقيام بعمليات متنوعة حسب نوعها.

مع شعار استخدم اللغة بكل طاقتها.

بدل أن تزيد أكوادك وتصرف مهارتك على شيء موجود أصلا كثير من يقول أنها تقلل من شأن المبرمج من قال لك

يا أخي لا تكن مثل أصحاب لغة السي وإصداراتها متحجر تريد أن تفعل كل شيء بيدك لا بل تقبله لصالحك.

ولكن ماذا ولو أننا مثل حساب تابع قوة لقوة خطر ببالنا مباشرة 8 أو أن نكتب الثابت ل Pi حتى في كل مرة نريده قمنا بفتح الآلة الحاسبة لإحضاره ولكن (خاصة لطلاب الدراسة) والذي لست منهم.

فأصبحت دوال البيسك لتشمل القيم العامة من اللوغاريتم بالنسبة لأي قيمة ونهاية بمقلوب نسب الزوايا ومنها تتحول نحو لغة تعليمية خاصة بالمسائل التعليمية مع زيادة في مجالات الأخرى

مميزات الدوال في الإصدار الجديد؟

١-تنوعها وشموليتها؟

٢-قدرتها على النمو؟

٤-فهرستها وتنظيمها(أصناف رئيسية مثل "الصف Math" وغيره)؟

٥-طريقة كتابتها البرمجية متميزة(ستجده فعلا هذه النتائج في حال كان القيمة "لانهاية كما تعلم أفضل من الطوفان")؟

٦-كل دالة هي كائن نستطيع القيام بمجموعة كبيرة من العمليات عليه؟

تشكيل التوابع وتركيبتها

تكن أهمية معظم اللغات البرمجية من آلية تشكيل دوالها أو إجراءاتها فكلنا يعلم أن الأشياء الأبرز ظهوراً في لغتنا ...

الأحداث	Events
الخصائص	Properties
المناهج	Methods
التوابع	Function
الأدوات والنوافذ	Tools
تعليمات أساسية	Statement

وعلى هذه الأشياء قائمة اللغة بشكلها العام وأن هذه الأغراض متداخلة ببعضها البعض من حيث الشكل والمضمون. (أعلم أنها كلها من شيء أي الإجمالي سيذهب إلى لغة برمجة)

ولعلي اقتصرت عليها من الناحية التعددية ولكن أشكال التوابع سوف تأخذ صورة التوابع أي أن الأحداث والخصائص والمناهج والتوابع هي كلها يمكن أن تندرج تحت إطار التوابع ((وهذا الكلام من وجهة نظري لم يقر به أحد.

ولكن ستبقى آلية تفضيل بين (التوابع التي ستقوم أنت بتصميمها وبين التوابع الأصلية في اللغة)

ستبقى توابعك والتوابع الجاهزة في سوية من المقدره مادام العمليات مقتصره على أوامر رياضية و تنسيقات نصية ولكن تصبح دوالك غير منطقية في حال الطلب من المكتبات الإضافية لماذا؟

أولا هم الذين يعرفون أنظمتهم وما تحتويه وكيفية الحصول عليه واعلم أن معظم دوالهم لهم فحسب أي مدعومة بشروحها لكن من الأفضل لنا كمبرمجين أن نضيف لنا ما يسمى شيان هما الوصف والاختيار أي من هرمية تشكيل التوابع أن نضيف شيء اسمه إل (Description)

"هذا التابع يستخلص اسم المستخدم" Des="Function Get User (By Val m as long)

فيظهر لنا في شريط Tool Tip Text

هذا التابع يستخلص اسم المستخدم

يعود هذا التابع باسم المستخدم فقط دون إمكانية تغييره أ

أليس كان أجمل ولكن ربما سيدرسون الفكرة في مستقبلهم التجاري ؟

بناء التوابع

الشكل البسيط:

إن الشكل الرئيسي لتشكيل التابع هو الإعلان عنه بالتعليمة

Function (Statement)

حيث كلمة Function هي التي تسمح لنا بكتابته وبعده نضيف اسم التابع

Function (اسم_التابع)

أوامر،

أوامر'

End Function

في حال كان شكله كالسابق سيكون أقرب ليكون إجراء Sub فكما تعلم نحن نشكل الإجراء بالشكل

Sub (اسم_الإجراء)

Msgbox "هذا هو الإجراء الأول"

End Sub

ولكنهم فرقوا بين الإجراء والتابع بأن التابع Function يستطيع أخذ نوع معين وإجراء عليه العمليات بهذا الشكل أي يقبل العمليات عليه فهو يعرف على أنه متغير

Function FunName () As String

FunName="السلام عليكم ورحمت الله وبركاته"

End Function

الحصول على قيم التابع،

Textbo~~x~~.Text =FunctionName

سيقول سائل لماذا تخط بين الإجراء Sub والتابع Function نعم إن الشكولين يستطيعا الوصول لنتيجة واحدة ولكن الثانية توفر من حجم المتغيرات وترتيبها واستخدامها أسهل نعم ما عليك معرفته لهذه المرحلة هو كيفية تشكيل التابع بمثاله الأخير مع أخذ بعين الاعتبار كيفية إدراج القيم الصحيحة إليه

والآن سنتجه إلى الشكل الجميل وذلك بإدراج الوسطاء إليه سنبدأ بمثال:

هل يا ترى سمعت بمعادلة من الشكل الثاني وحساب قيمة الدلتا

فكلنا يعلم أن لمعادلة الدلتا ثلاث وسطاء (ا و ب و ج)

$$اس^2 + ب س + ج = ٠$$

$$\blacktriangle = ب^2 - ٤ ا ج$$

ولو أردنا أن نصنع تابع لحله ليس من المعقول أن نخرج أن نجعل المبرمج كمستخدم باستخدام

أدوات الإدخال Input Box

بالتأكيد نريد حساب جذر الدلتا

Function Delta(By Val A As Long , By Val B As Long ,By Val C As

long) As Long

Dalta=Math.Pow(B,2)-4*A*C

End Function

طبعا إن عملية استدعاء هذا التابع ستكون

Call Dalta(10,5,2

و في حال وأنت نسيت أن تكتب الوسيط الصحيح فإن التسطير تحت هذا التابع سيكون من نصيب ذلك التابع:

إن آلية التحكم مرتبطة وفق خوارزميات نعم. لن تضع ميكروسوفت جميع التوابع في العلم وتقول تفضل هذه تفيدك ولكن هي تنتقي الأصعب والأكثر شهرة .
ولكن كثير من التوابع لا تتأقلم معنا ولكن يعني لو جميع العلاقات الرياضية في العالم مرتكزة على العمليات الحسابية الأربعة طبعا سيتطلب ذلك تعقيد في تشكيلها فعلينا أن نعود إلى لغة الأسمبلي .

فتم إضافة ما يسمى اللوغاريتم والجذور وهلم جرى من التوابع للوصول للحساب المساحات ولكن تكمن أهمية التوابع من حيث آلية التنسيق فمثلا سأوضح هنا بعض الفر وقات.

فلنأخذ برنامج الفلاش بعد الإصدار السادس ولغة الفيچوال أنت
-لوجدت الفلاش يدعم الأدوات كما في الفيچوال وإذا قلبت في دواله لوجدت أنه يحوي نفس دوال أنت في صنف الرياضيات ونفس الدوال في الصنف السلاسل النصية
--يجب أن يصبح مفهوم البرمجة جديد أي أدركت مايكروسوفت هذا المعنى

ولكن أود أن أوضح أن كثير من دوال البرمجة مخفية عن التعرف فمن هذا الذي يود أن يغوص في ثنايا ال Help ليجد مثال بسيط وشامل لعمل ذلك التابع بالشكل الذي أرجو أن يصل إلى الغاية بأقصر طريق

لعل أهم الأشياء في تشكيل التابع هو أنه تم إضافة المزيد من التعليمات الأساسية أي قاموا بإلغاء تعليمات أساسية كتعليمية فتح الملفات و تغيير أسمائها ولعلك تلاحظ أن اللغة بدأت ترتيب تصميمي أي إن كل التعليمات التي ستكون مدعومة هي تشكيل التوابع بها ومادون ذلك فهي كما نوهنا إليه

الشكل المتعدد

ليس لكل قانون وسائط نوجد قيمته من خلاله قيمة فمثلا حساب مساحة المستطيل .
سط = قيمة = عمليات على الوسائط(التي سنستخدمها للوصول إلى مساحته)
ومنه كان الأفضل لهذه اللغة أن تقدم ذلك أيضا كي يسهل على المستخدم فهم ذلك التابع واستخدامه ومن هنا نستطيع كتابة الشكل المتعدد

Function المستطيل_مساحة

ولكن علينا أحيانا أنه نوجد وسطاء ليس من الضروري استخدامها أي نستطيع تجاهلها بالانتقال إلى الوسيط الآخر فمثلا تابع بوسطاء لم تدخل ووسطائه كاملة فلن يسمح لك المترجم أن تنفذه حتى تصححه ومع هذا وذاك يتطلب حل هذه الأزمة في حال أن المستخدم لم يكتب الوسيط

FunctionDF(A

Calldf () خطأ لان لم تستكمل بوسطاء بشكل كامل

والحل الآن:

بداية سأذهب إلى التالي فعلى سبيل المثال تذكر معي عبارة الإسناد التالية

```
Dim FName As String = "nameuswr"
```

إنها من الجديد التي جاءت به مايكروسوفت وهو اختصار أن تستخدم في برامجك وأيضاً المبرمجين كانوا قد قدموا عبارة

```
FunctionFG (Optional ByVal FName As String="محمد")
```

هنا حلت المشكلة التي ذكرتها حيث أنه إذا لم يدخل المبرمج الوسيط سيأخذ القيمة الافتراضية التي مررت إليه مباشرة وهي "محمد" ومن هنا سنتجنب الأخطاء الكثيرة التي لا نريد أن تقع بها حيث تعليمة **Optional** هي التي حلت المشكلة للقيم الافتراضية.

ومن هنا ضع هذه القيمة في حال تخشى إدخلات المستخدم المزعجة أحياناً
ملاحظة: أعرفت لماذا غيرت طريقة الإسناد وأضاف عليها إسناد القيم مباشرة

لا يوجد خطأ، Call FG ()

لقد تم جعل المحرر يمرر الوسيط بالقيمة مباشرة (By Val) وإذا أحببت أن تجعلها بالمؤشر فعليك أن تغيرها بنفسك لأن تم على بشكل ملحوظ تحسين التمرير بالقيمة في قيم الإعدادات الكبيرة

الجديد في تشكيل التوابع والإضافات

التعليمة (Return)

لقد قدم لنا المبرمجون والمبرمجات في لغتنا الحبيبة تعليمة (العودة) للقيمة وليس حق العودة لفلسطينيين فهم كرماء معنا في الإصدار السابق كان هناك شيء غير مفهوم بكتابة التابع فمثلاً

```
FunctionLname( ByVal NumberName As Short) As String
```

```
If NumberName=1 Then
```

```
Lname()="مستحيل"
```

```
Else
```

```
Lname()="غير مقبول"
```

```
End If
```

```
End Function
```

فلو رجعت لذلك المثال لوجت هل من المعقول أن نكتب التابع ولا نمرر له وسيطه طبعاً هذا الاستعمال فقط داخل التابع ولم يتم إلغائه ولكن تم المجيء بالتعليمة **Return** بالعودة بقيمة وهي تستخدم بالتوابع فقط على ما أظن أي لم أرها استعملت بغيرها وفائدتها الاختصار وإزالة اللبس عن ذلك من وجهة نظري على الأقل وهكذا يصبح شكل التابع السابق على الشكل التالي:

```
FunctionLname( ByVal NumberName As Short) As String
```

```
If NumberName=1 Then
```

```
Return "مستحيل"
```

```
Else
```

```
Return "غير مقبول"
```

```
End If
```

```
End Function
```

أي في حال وجدت هذه التعليمة وورائها قيمة معينة ستمرر مباشرة وتصبح هذه القيمة هي قيمة التابع

ملاحظة هامة: إن بعض المبرمجين صرح على أن هذه التعليمة جاءت لاختصار ولكن قلت

إضافات نستحق أن نذكرها

دمج التركيبات مع التوابع؟

كثيرا ما لاحظت وأنت تستخدم الدوال الجاهزة ذات الوسطاء ترى أن العديد منها تحوي على قيم ثابتة تبعد اللبس في الإدخال فلذلك نستخدم التركيب (Enum) حيث نستطيع من خلالها بناء مجموعة من القيم التي قد تكون من الممكن أن نعتبرها من الثوابت وهي من النوع Long وإليك مثال على استخدامها أي أنت من خلالها تقوم ببناء العديد من المركبات ذات القيم المناسبة

```
Enum Fname
    Fat16
    Fat32
    Ntfs=64
```

End Enum

```
Function Recod (ByVal Fname As Fname) As String
```

```
    Select Case Fname
```

```
        Case Fat 16
            MsgBox "نظام القديم"
            .....
            .....
```

End Select

End Function

دمج التوابع في المناهج

عد بالذاكرة إلى التركيب من نوع User وهي التعليمة Type التي هي عبارة عن مجموعة تراكيب من أي نوع كان من البيانات وكانت عبارة عن فرق التركيبات Enum عن Type هو أنه النوع Type يدعم كل أنواع البيانات وأما Eunm هي فقط من النوع Long .

```
Private Type FG
```

```
    D As Long
```

```
    F As String
```

End Type

ستقول لي ما هذا الكلام الجديد.... سأقول للمبتدى في الإصدار الجديد الكلام في الأسطر السابقة

لا تلزمك واقر مادون والسبب لأنه لقد ألغى و استبدلوا المبرمجين التعليمة السابقة في الإصدار

الجديد بنوع آخر أشبه بما يسمى الصنف أو البناء وهو Structure

وهذا التعليمة بديلة عن Type ولكن بتغيرات بسيطة بإضافة تعليمة الإسناد

```
Enum country
```


الشام
العراق
الجزيرة_العربية
مصر
المغرب_العربي

.....

.....

End Enum

Public Structure CollUser

Const UserName=" NewPerson"

Dim Name As String

Dim ega As Short

Dim name2 As String

Dim yourcountry As Country

'Dim f as short=50 هنا خطأ لأنه ممنوع إدخال القيمة للمتغير ضمنها

End Structure

استدعاء القيم منه والإسناد

Dim CallCollUser As CollUser

CallCollUser.Name="محمد"

CallCollUser.Ega=60

CallCollUser.Yourcountry=Country.الشام

MsgBox CallCollUser.UserName

والمزيد ستراه يمكنك أيضا إضافة التوابع ضمن الصنف نفسه وهذه الميزة من إحدى الميزات الهامة التي تجعل هذه التعليمة الأساسية أشبه بأن يكون صنف.

Public Structure CollUser

Function ShowMyBox () As String

Dim NameS As String

.....

.....

.....

.....

End Function

End Structure

دوال التحويل

Conversion

لقد تم تصغير هذا الصنف لدوال التحويل الخاصة بالأعداد وتم نقل التوابيع الأخرى لصنف آخر وتم الاحتفاظ على ٧ دوال قابلة لزيادة وسنشرحها حتى يتثنى لنا الانتقال إلى صنف آخر.....

الدالة (ErrorToString)

هذه الدالة بدل الدالة Error و Error\$ في الإصدار القديم ولكن هذه دالة ترجع وصف الخاص برقم الخطأ أي ترجع رسالة الخطأ النصية لرقم الخطأ المتوقع أي أثناء كتابتك برنامج معين وقمت بوضع حلول لمعظم الأخطاء المتوقعة ولكن حتى يحدث خطأ لا نستطيع تحديده يرجع رسالة الخطأ الخاصة به (مثل Debugs) ونلخصه بالتالي :

إن جل أرقام الرسائل ٠ إلى مجال (القيم العددية) تعود بوصف خاص فمثلا وصف الخطأ ذي الرقم ٦٣ "تخبرنا بخطأ برقم السجل" أي كل رقم له وصف خاص ومن هنا يمكننا كتابة سجلات أخطاء لبرامجنا وفي حال أخبرك المستخدم على أن برنامجك يحوي أخطاء أخذت سجل الأخطاء وراجعت أرقامها ومن ثم طورت برنامجك وتمنع حدوث مثل هذه الأخطاء وهذا مثال يظهر لك بعض الرسائل الممكن حدوثها :

```
ListBox1.أضف أداة قائمة اسمها '
Dim s As Integer
For s = 1 To 5000
ListBox1.Items.Add (ErrorToString(s).ToString + s.ToString)
Next
```

الدالة (Fix)

في معظم الأحيان نُحبذ لو نُدور العدد العشري إلى العدد الطبيعي دون تقريب أي العدد (٥٢٤,٩) لانريد تقريبه مثلا ليصبح (٥٢٥) فتقوم هذه الدالة بنزع الفاصلة وما يليها مهما كانت القيمة سالبة موجبة وأي نوع من البيانات كان المتغير بالكلام العام. وهذا مثال عليها:

```
Dim X As Integer
X=14*Rnd ()
X =Conversion.Fix (X)Output-X
MsgBox(X)
```

الدالة (Hex)

جاء دور الحديث عن دوال التحويل من أنظمة العدد العشرية إلى أنظمة العد الست عشري

كما نعلم في معظم محررات البرامج (الكر اكر) تخرج لنا المحررات البايتات المأخوذة من الملف بصيغتين ست عشرية وأخرى بشيفرة ال ASCII المقابل لها فأما الست عشري فهي خاصة بمن يفهم لغة الأسمبلي ومثابه فيعرفون مكان المقطع أو المكس طبعا نظام الست عشري يبدأ:

1 2 3 4 5 6 7 8 9 A B C D E F

وطبعا العدد الذي يجب علينا تحويله يجب أن يكون أولا طبيعيا (وليس سالباً) وأقصى حد أن يكون من النوع Long
وكي تحفظ متغير بقيمة ست عشرية استخدم البادئة (&H) ومن ثم أضف العدد الذي تريد فتلقائياً تحوله اللغة إلى قسمة عشرية في حال الاستعمال ولكن عندما نحوله إلى ست عشري فإعلم أنه يتحول إلى قيمة نصية حتما

```
Dim X As Long = &HEFF
```

(هنا تلقائياً حولته اللغة إلى النظام العشري وMsgBox X & "الست عشري")

هنا حولته اللغة إلى النظام " &MsgBox Conversion.Hex(X) " (الست عشري)

(الست عشري)

هنا خطأ نوع المتغيرات إذ أنه تحول إلى (X=Conversion.Hex(x))

نوع بيانات آخر

الدالة (Oct)

طبعا من الأنظمة المشهورة أيضا يوجد النظام الثماني أي نبدأ العد من ١ إلى ٨ فقط إذ تحول من النظام العشري إلى النظام الثماني ولكن لا نحتاج للعكس وكى نحفظ متغير بقيمة ست عشرية نستخدم البادئة (&O) ومن ثم ندون العدد

```
Dim X As Long = &O5252
```

(هنا تلقائياً حولته اللغة إلى النظام العشري وMsgBox(X & "العد الثماني")

هنا حولته اللغة إلى النظام " &MsgBox(Conversion.Oct(X) " (العد الثماني)

(العد الثماني)

هنا خطأ نوع المتغيرات إذ أنه تحول إلى (X=Conversion.Oct(x))

نوع بيانات آخر

الدالة (St)

تقوم هذه الدالة بعملية تحويل الأعداد من النوع Object إلى قيمة نصية أما فائدتها الجوهرية نستخدمها في قواعد البيانات إذا أردنا أن نحول عمر الشخص إلى قيمة نصية ونضيفها إلى قواعد البيانات ولكن نافذة Format



MsgBox (8.)

على المتغيرات قدمت لنا الأمر ToString ولكن هذه الدالة موجودة على كل حال والفرق عن ToString أنها لا تحول إلا الأعداد ولا تقبل بارامترات (وسطاء) من أنواع غير الأعداد

```
MsgBox(Conversion.Str(851455.52)) 'Output851455.52
'MsgBox(Conversion.Str(851455.52)) 'Output خطأ جميل
```

الدالة (Val)

وهي من الدوال الشائعة الاستعمال فأحد استخداماتها أنها تمنع حدوث أخطاء أثناء إدخال البيانات

فهي تقوم بتحويل السلاسل النصية إلى قيم عددية (مثلا تحول Null إلى ٠)

```
MsgBox(Conversion.Val("5f45f ")) 'output 5
```

دوال الموارد

Global

بعيدا عن التعليمات Global التي كانت مدعومة في الإصدارات ما قبل الإصدار السادس وعندما جاء الإصدار السادس حيث كانت تستخدم مع دوال الـ API وتجعل التابع عام يرى في كل مكان مثل التعليمات Public ومن ثم تحولت في الإصدار السادس إلى صنف شامل لكل الموارد الخاصة باللغة لكن لم تكن تعمل بشكل صحيح وكانت أحد الأخطاء الموجودة فيها ولكن في إصدار السابع المسمى (CLR).. أصبحت تعبر عن شيء مهم وهو أن بأي لغة برمجية تم برمجته والإصدارات المتعلقة به (لعلها أول خطوة في إطار الـ Framework ومع ذلك لم يتم دعم إلا بضع الدوال وهي نوع لغة السكريبت المتبعة و إصدارها ونشرها وأحببت أن أضيف هذا الصنف رغم عدم أهميته

الدالة (ScriptEngine)

التعبير يصعب شرحه من ناحية الترجمة ولكن نستطيع إعطائه صورة وهو المترجم التي تم كتابة التعليمات النصية ضمنه من حيث التعليمات المكتوبة هل هي Vb أو Vc أو C#

```
MsgBox (Globals.ScriptEngine) 'output Vb
```

الدالة (ScriptEngineBuildVersion)

الدالة (ScriptEngineMajorVersion)

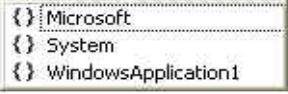
الدالة (ScriptEngineMinorVersion)

تلك الدوال الثلاثة السابقة تعطي إصدار المترجم الذي تم كتابة التعليمات النصية ضمنه من حيث التعليمات المكتوبة وهو لا يتعلق بإصدارات برنامجك أي ثابتة بالنسبة لإصدار اللغة التي تعمل عليها

```
MsgBox (Globals.ScriptEngine + " " &
Globals.ScriptEngineMajorVersion & "." &
Globals.ScriptEngineBuildVersion & "." &
Globals.ScriptEngineMinorVersion)
```

ومن هذا الصنف فنحن أمام دوال ربما تعطينا عدد التوابع المستخدمة والتعليمات و القيمة الكمية للمتغيرات وربما نرى المزيد إن شاء الله
 أما في الإصدار ٢٠٠٥ فلقد تحولت الصنف Global إلى الصنف الرئيسي للغة حيث أصبح هو الموجه الرئيسي له وستوضح الصورة التالية ذلك

```
Private Sub Form1_Load(ByVal sender As System.Object
Global.|
End
```



حيث الأصناف الرئيسية (الجزر الرئيسي أصناف وموارد اللغة تدرج منها

الدوال الرياضية

Math

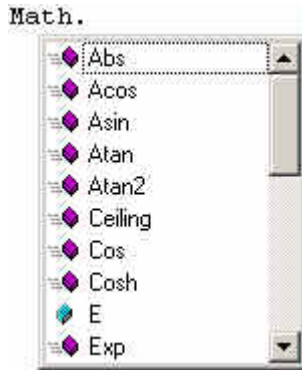
تحت إطار العمل القديم أوجدت مايكروسوفت العديد من أماكن التي تستطيع من خلالها البدء في عمليات التطوير.

مثال: تابع اللوغاريتم يحتاج لمجموعة متنوعة من الأسطر لتحويله أساسه من العدد النيبري إلى العشري فماذا لو أردنا تحويله إلى أكثر من ذلك ولكن هنا قدمت مايكروسوفت حلّ تراه هي من وجهة نظرها يكفي ولكن بقيت لغتنا العزيزة تعمل على خلفية نظيفة فصحيح عن قدراته الأمنية الخارقة والشبه المعقدة وأغراضها التوجيهية ودعم الوراثة إلى أنها لم تكن إلا أداة تطوير تجارية من الناحية البرمجية فمثلا لا تنتظر من برنامج عمل المعجزات أي بيئة العمل في إطار العمل أشبه من عملية تجسس من خلالها على خصوصياتك أي لا تحلم أن تصنع آلة حاسبة وتريد منها أن تعمل بشكل مباشر على نظام (مليين يوم) بل على مستخدمها أن يدفع تقنيا أو عمليا بعمليات تنزيل ملفات إضافية تحتاج إليها

سوف نقف عن التكلم عن أشياء لا نهم المبرمج ونبدأ بشيء اسمه ما هو الجديد في دوال هذا الصنف لو أردنا الكلام عنه لأنقصناه حقه فلقد إضافة ما فوق ١٥ تابع جديد وأجرت عمليات تغيير إما اسمية أو تطبيقيا على بعض ما تبقى منها ولكن في صريح ذلك كله تبقى الإشارة لفروق واضحة في الإصدار الجديد وإن التعديل الهرمي على اللغة سيأتي أيضا على التوابع فمثلا لم نعد نرى علامات الدولار لمعظم التوابع التي اعتدنا على كتابتها في لغتنا السابقة وذلك لمعالجتها برمجيا بالأصل وهذه هي صورة هذا الشيء الجميل من ذلك الصنف الأم لدوال الرياضيات

ولكن بقيت أشياء لم تحافظ مايكروسوفت عليها من باب الترتيب وعدم الضياع بين مئات التعليمات فأوجدت شيء تجعها فيه كما في الإصدار القديم وهو الصنف الحاضن لها أي مثلا أنت تريد أن تتعامل مع شيء اسمه الجيب فعليك أولا الدخول إلى شيء اسمه الصنف

الحاضن لمجموعة علاقات الرياضية وهو



أحببت أن أضيف هذه الصورة لعلها تقي بالغرض وإضافات للصنف أيضا ما يسمى ثوابت تتعلق به من الناحية العملية وإليك التفصيل

البداية مع الثوابت التي تم إدراجها ضمن الصنف الجديد وهما ثابتان الأول

(PI) الثابت

وهي القيمة التقريبية بين محيط الدائرة وقطرها ونرمز لها نحن π ونعرفها $3.1415926535897932384626433832795028841971693993751058209749445923078164062862089986280348253421170679821480865132823066470938446095505822317253594081281$ وهي ثابت فقط فبدل أن تكتب القيمة السطر

Const Pi=32.14563665

وكما تعرفون دائما أن معظم المبرمجين لا يحفظون قيمة Pi أو أنهم بالأصح يفتحون آلة الحاسبة بالجهاز ويحضرون القيمة الحقيقية فكانت هذه الدالة ضرورية وهم قاموا بإدراجها تكتب

Math pi

(E) الثابت

كثيرون من يجهلون ما هو العدد النيبيري والكثير أيضا يعرفه . وكان في معظم الدراسات يكتب مساوي $2.7182818284590452353602874713526624977572470966585924672$ (e).

وهو يمثل القيمة العددية للعدد النيبيري الذي لو غارتميه $e^1 = 2.7182818284590452353602874713526624977572470966585924672$ هنا تم إنجاز الثوابت وننطلق نحو التوابع ونبدأ

(Abs) الدالة

تعيد القيمة المطلقة للقيمة الممرر إليها ومن أي نوع كان من البيانات الحسابية
المثال:

MsgBox(Math.Abs(-3.55655))

وهذا الكود الأولي يوضح عمله لو كان ليس موجودا:

```
Function AbsNumber(ByVal number As Double) As Double
    If number < 0 Then
        AbsNumber = number * (-1)
    Else
        AbsNumber = number
    End If
End Function
```

(Ceiling) الدالة

وهي تعود بالحد الأقصى للقيمة المدخلة إليها. يعني تحول أي قيمة بعد الفاصلة ما عدا الصفر إلى رقم تضيفه على رقم أحادها وعند القيمة أصغر من الصفر تحولها إلى أصغر قيمة قريبة يعني تلغي الفاصلة فحسب وهي من الجديد المضاف.


```
MsgBox(MathCeiling(39.0))
```

الدالة (Floor)

وتعود بأصغر قيمة للعدد العشري. والتي تتمثل بإلغاء فاصلتها وما بعدها أي عكس سابقتها. تفيد هي والتي قبلها في بعض المعادلات الرياضية
مثال :

```
MsgBox(MathFloor(10.1))
```

هاتان الدالتان تفيدان في حساب بعض أنواع الارتياح و بعض علاقات الرياضيات

الدالة (Pow)

هذه الدالة تعود بأساس أي رفع القوة وهذه طريقة استعماله ستقول لي لماذا أضافت المبرمجين هذه الدالة حتى يسهل عليهم حساب الجذر النوني أي $\sqrt[n]{x}$ من أي مرتبة لأن مصطلح Power هو مصطلح شائع في اللغات الأخرى وهذه مثال عليه.

```
MsgBox(MathPow(10,10)*100000000000)
```

ويمكن القول أنها تشبه هذا الرمز (^)

الدالة (Sqrt)

الدالة جديدة فقط اسميا أي هي نفس الدالة Sqr والتي ولدت مع لغة البيسك و وهذه الدالة موجودة في كل الإصدارات وظيفتها إيجاد الجذر التربيعي للقيمة من نوع Double ولها وتأخذ الشكل التالي

```
Sqrt(Number As Double) As Double
```

وطبعاً يعني تقبل جميع القيم التي تراوح فقط في المجال الموجب لأن الجذر التربيعي لعدد سالب مستحيل وهذا المجال

4,940.60640841247 e+324

وهذا مثال يوضح ذلك

```
Private Sub Form_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
الكود
MsgBox(Math.Sqrt(9)) 'output3
End Sub
```

الدالة (Sign)

الدالة جديدة فقط اسميا أي نفس الدالة Sgn وهذا التابع أيضا من التوابع القديمة التي نشأت مع البيسك وهو يعيد إشارة العدد الممرر له على شكل قيم . فإذا كان إشارة العدد سالب يعيد العدد - ١ وإذا كانت إشارة العدد موجب (يعني بدون إشارة) يعيد العدد ١ وإذا كان العدد يساوي الصفر عنده يعيد الصفر وأحد أماكن استخدام هذا التابع هو حساب الدلتا لمعادلة من الدرجة الثانية وهذا مثال يوضح عمله

```
Private Sub Form_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
    MsgBox(Math.Sign(562)) 'Output1
    MsgBox(Math.Sign(-33)) 'Output-1
    MsgBox(Math.Sign(0)) 'Output0
End Sub
```

الدالة (Log)

وهذه الدالة تعيد اللغاريتم الطبيعي لعدد بالنسبة للأساس العدد النيبري وليس العشري كما هو الحال في الآلة الحاسبة وسوف أضيف التابع الإضافي لحصول على لغاريتم العدد للأساس ١ التي أدخلته مايكروسوفت بإصدارها الجديد وكما نعلم فيجب على العدد الذي يدخل لها أن يكون أكبر من الصفر وإلا رسالة الخطأ سوف تظهر لك طبعا تعرف لماذا. لأن مجال تعريف التابع اللغاريتم منطلقه (+ح) * ومستقره (ح) وهذا مثال يوضح عمل هذا الدالة -ولكن في الإصدار أنت قدمت مايكروسوفت شيء جديد حيث نستطيع حساب اللغاريتم بالنسبة لأي أساس دون أي جهد طبعا كلامنا السابق صحيح إذا لم ندخل أي من البارمترات وهذا شكل التابع الجديد

```
Math.Log(D as double , newbase as double)
```

ومن هنا نجد أنه من السهل معرفة المقصود وهذا مثال يوضح الغرض بشكل بسيط

```
MsgBox Math.Log(10, 10)
```

الدالة (Log0)

هذه الدالة إن دلت على شيء اسمه السرعة فهي من الناحية العملية إختصار إلى الدالة السابقة ممرر إليها أساس العشري ولكن لا أدري لماذا قاموا بوضعها أتوقع أنها قاموا بوضعها قبل سابقاتها ومن ثم قاموا ببرمجة سابقتها وتركوها وهذا مثال يدل عليها:

```
MsgBox Math.Log(10)
```

الدالة (Exp)

وهذه الدالة أيضا قديمة مثل سابقتها وتقوم بإرجاع العدد الطبيعي للعدد النيبري مرفوع إلى قوة وهذا التابع

مكمل لتابع الذي قبله وفي بعض الأحيان يسمى في تابع عكس لتابع اللغارتيمي ومع العلم أنه لايقبل قيمة أكبر من ٧٠٩,٧٨٢٧١٢٨٩٣ وإلا فرسالة الطوفان سوف تظهر لك ولكن في المبرمجين في شركة مايكروسوفت كشفوا الأمر وإستبدلوا رسالة الطوفان بشيء إسمه اللانهاية ال



هذا المثال الذي رسم هذا الصندوق

`MsgBox(Math.Exp(780), MsgBoxStyle.OKOnly)"`)

الدالة (Round)

أما هذه الدالة فهي جديدة على البيسك و وحتى على الإصدارات التي سبقت الإصدار السادس وظيفتها مفيدة ولكن يجب أن نتعرف على شكلها

`Round(Number, NumDigitsAfterDecimal)`

حيث القسم الأول يمثل عدد ما يحوي فاصلة بعدها عدة أرقام فنقوم هذه الدالة بإظهار عدد الأرقام (القسم الثاني الذي يمثل عدد هذه الأرقام) التي يجب أن تظهر بعد الفاصلة وكما نعلم فإنها شبيهة بدوال أخرى ولكن تقوم بوظيفة مغايرة ومع العلم أن أهميتها كبيرة في حساب بعض القوانين الفيزيائية وبشكل خاص. حيث تفيد في تقريب الأعداد وإذا قمنا في حذف الوسيط الثاني فنقوم بدورة التقريب الأعداد بعد الفاصلة (أكبر من ٥ تضيف ١ للعدد الذي يليه على يساره وإلا تقوم بحذفه):

```
Private Sub Form_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
MsgBox (Math.Round(56.656262653)) 'OutPut-
56.656
MsgBox(Math.Round(155.5)) 'OutPut156
MsgBox(Math.Round(155.5)) 'OutPut156
MsgBox(Math.Round(155.3)) 'OutPut155
End Sub
```

الدالة (Min , Max)

هنا جاء نوع المقارنة وهذه الدالتان من الجديد الذي تم إضافته في الإصدار الجديد وهذان الدالتان نفس الاستخدام ولكن تأتي بنتيجة معاكسة للأخرى المهم أنهما تقومان بعمليات المقارنة بين متغيران من النوع نفسه (بزمرة المتغيرات العددية) وتعيد الدالة (Min) القيمة الصغرى بين المتغيران ١ وأما الثانية تعيد القيمة الكبرى من العددين .
المثال:

```
Private Sub Form_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
    Dim number1 As Long = 1000
    Dim number2 As Long = 2005
    MsgBox(Math.Max(number1, number2)) 'output 2005
    MsgBox(Math.Min(number1, number2)) 'output 1000
End Sub
```

الدالة (IEEERemainder)

وهذه الدالة جديدة أيضا فهي تحل محل التعليمة الأساسية {Mod} أي ترجع الباقي من أقرب قيمة لحاصل قسمة عددين أي $3,33333 = 3 \div 10$ فهذه الدالة تقوم بعملية القسمة لأقرب عدد طبيعي وليس صحيح وترجع الباقي أي $3 = 3 \div 10 +$ الباقي ١ أي القيمة التي ستعود بها هي ١ وشكل التابع كالتالي

```
Math.IEEERemainder(x As Double, y As Double)
العدادان x, y هما وهما القاسم والمقسوم عليه
وهذا مثال
```

```
MsgBox(10 Mod 3) 'output 1
MsgBox(Math.IEEERemainder(10,3)) 'output 1
```

الدوال (Sin, Cos, Tan)

جاء الدور للحديث عن دوال النسب المثلثية الشهيرة. وجميع هذه الدوال هي دوال لحساب النسب المثلثية (الجيب. التنجيب. الظل.) على الترتيب وقد إضافة ميكروسوفت في الفيچوال بيسك ٧ توابع إضافية أخرى. ولكن هذه النسب في الفيچوال بيسك تستخدم نظام الراديان وكما نعلم أنه يوجد أنظمة عديدة من أشهرها (الراديان. الدرجات. الغراد) ونعلم أن قياس القطاع الزاوي للقطعة المستقيمة تقابلها في الأنظمة الثلاثة (٢٠٠_١٨٠_٣,١٤١٥٩٢٦٥٣٥٨٩٧٩) على الترتيب. طبعاً لا يغيب عن ذهنكم أنها توابع تساعد في حل المسائل الرياضية إذ تستخدم في معظم القوانين الفيزيائية والرياضية وبإذن الله سوف أضيف التوابع الأخرى المصممة شخصياً والتي أدخلتها مايكروسوفت في إصدارها الأخير. وإليكم هذا المثال من أجل تحويل بين الزوايا مع العلم أن هناك دوال لا تعطي المرجو منها في كل مرة وقد حُسن أدائها في الإصدار الجديد

```
Private Sub scn()
Const De=180
MsgBox(Math.Cos(PI/De*60)) 'output 0.5000001
0.7071067811865475
MsgBox(Math.Tan(PI/De*45)) 'تقريباً ١
MsgBox(Math.Sin(PI/De*30)) '0,5
End Sub
```

الدالة (Acos)

الدالة (Asin)

الدالة (Acos)

فهذه الدوال جديدها مفيد ومن القوانين تزيد سمعنا بأن لكل زاوية جيب وتنجيب وظل وتظل ولكن ان لكل (جيب) زاوية ولكل (تنجيب) زاوية.
يعني تدخل له قيمة التنجيب فسوف يخرج لك قيمة الزاوية وطبعا جميع القيم هي في الراديان فكما تعلم أن جميع القيم في جميع الأنظمة هي واحدة فمثلا قيمة التنجيب ل0,5 هي لزاوية 30 و 0,5235987750598829887307710723054658 هي بالراديان فلا فعليك إذا صناعة تابع لهذه المهمة راجع كيفية تشكيل التوابع

```
Function DeAngle(ByVal Angle As Decima) As Long
    DeAngle = 180 * Angle / Math.PI
End Function
MsgBox (DeAngle(Math.Asin(0.5))) '37
MsgBox (DeAngle(Math.Acos(0.5))) '120
```

يعني الأمر ليس معقد فالأمور بسيطة ويلزم هذا التابع في كثير من معادلات الرياضيات أو الفيزيائية مثل (محولة أحادية الطور)

ونفس الأمر لما تبقى من التوابع Atan يرجع قيمة زاوية بعد إدخال ظل تلك الزاوية ومن هنا يمكنك حساب تلك القيم وحساب قيمة التظل لزاوية فهي تساوي مقلوب الظل أو التنجيب / الجيب تظل(يه) = 1 / ظل(يه) أو تجب(يه) / جب(يه)

الدالة (Cosh)

الدالة (Sinh)

الدالة (Tanh)

في البداية تراجعت عن كتابة هذا الدوال خشية أن أخوض بما ليس لي علم فسألت أحد الاخوة عن المعنى العلمي لها: هذه تفيد في بعض العمليات الحسابية والفيزيائية وهي من العلاقة
الزاوية - الزاوية

$$\text{Sinh(الزاوية)} = (e^{\text{الزاوية}} - e^{-\text{الزاوية}}) / 2$$

ويسمى الجيب القطعي

$$\text{Cosh(الزاوية)} = (e^{\text{الزاوية}} + e^{-\text{الزاوية}}) / 2$$

ويسمى الجيب القطعي

أما الظل القطعي

$$\text{Tanh(الزاوية)} = \text{Sinh(الزاوية)} / \text{Cosh(الزاوية)}$$

والتظل القطعي مقلوب الظل القطعي

```
Dim f As Double
f = Math.Pow(Math.E, 2) - Math.Pow(Math.E, -2) / 2
MsgBox(f)
MsgBox(Math.Sin(2))
```

هذا المثال صدق كلامي

ملاحظة:

صحيح أنه يوجد كثير من التوابع التي يمكنك إضافتها مثل تابع التظل والمتمم والقاطع معتمدا على توابع الموجودة فتابع التظل هو مقلوب تابع الظل ويمكنك إضافة ما يسمى الجذر النوني كما شرحنا وكثير من العلاقات التي يمكنك من إضافتها في مكتبة أو ترسلها إلى فريق تطوير اللغة ليتم إرفاقها في إصدارات لاحقة.....

أما دوال الإصدار ٢٠٠٥

الدالة (BigMul)

تقوم بحاصل الضرب بين عددين a, b من النوع `Int32` وتعطي كامل الأرقام من النوع `Long`

```
Dim n1 As Integer=Int32.MaxValue
Dim n2 As Integer=Int32.MaxValue
Dim MaxNumber As Long
MaxNumber = Math.BigMul(n1,n2)
MsgBox(MaxNumber) 'output4611686014132420609
```

(TrunCate) الدالة

وتفيد هذه الدالة بإرجاع القيمة الصحيحة من العدد المضاعف أو العشري فهي تحذف كل عدد بعد الفاصلة دون تقريبه ومثال على ذلك

```
MsgBox(MathTruncate(35.6)) output35
```

الدوال الرياضية القديمة

VbMath

الدوال الموجودة في الصنف VBMath؟

لقد أردت أن أبدأ بالشيء الأقرب للغتنا الجميلة القديمة فهذا الصنف يتضمن حالياً فقط دالتين يعبران عن دالتين من زمن البيسك الأولى وهاتان الدالتان.

الدالة (Rnd)

كما نوهنا أولاً إن عمليات ترتيب الأصناف تتطلب عمليات تنسيق لها لذلك علينا تذكر عملياً أن كل دالة لها صنف رئيسي تابع لها وكلما حددنا الصنف الأعلى كان أسرع على المترجم إيجاده وبالتالي زيادة عمليات المعالجة نحو ١٠% تقريباً ولكن الدالة Rnd هي دالة غريبة

هذه الدالة أيضاً قديمة وقد تراها في نظام

Dos

في بعض الأحيان ووظيفتها تقوم بتوليد سلسلة من الأرقام المحددة بقيم أنت تحددتها ودورها مهم فأنت باستخدامها تستطيع توليد مجموعة ألوان تغير من لون الخلفية عند نقرت كل زر ولكنها تأخذ قيمة أولية ثابتة بالنسبة للقيمة المدخلة ويمكن تغييرها بواسطة استخدام أحد التوابع المذكورة وهي تعمل كالتالي أولاً عندما ينفذ الإجراء أول مرة تقوم بتخزين قيمة معينة في الذاكرة وهي ثابتة لكل عدد وعند تنفيذ الإجراء مرة أخرى تقوم عبر خوارزمية محددة بتوليد عدد جديد اعتماداً على العد الثابت المخزن ولكن عند إنهاء البرنامج المكان الذي حجزته أصبح فارغاً فلذلك تعود وتكرر نفسها في المثال اللاحق ولكن في شكله العام

Rnd([Number])

حيث الوسيط الممرر إليه يشمل قيم معينة مستخدم (-.*./) من أجل ضبط القيم ولكنه في بعض الأحيان يظهر العدد ومع فواصل ولتجنب ذلك نستخدم إما التابع الذي قبله بعد حذف وسيطه الأول أو تابع من توابع التحويل وتذكر أنه في كل مرة سينفذ البرنامج سيعطي قيمة نفسها بآداء الأمر لكل عدد قيمة ثابتة ستظهر له وإليك المثال التالي يشرحه شرحاً مفصلاً

Dim s, dAs Integer

MsgBox (Rnd() * DateAndTime.Timer - 500) ' هنا سوف يظهر

الرقم ليس نفسه في كل مرة

MsgBox (Rnd) 'output.7055..

For s=0 To 3

MsgBox(Int(Rnd() * 5)) ' وأكبر من ٠ جميع الأعداد الأصغر من

الخمسة

Next


```

Ford=0      To 5
MsgBox(MathRound(Rnd) * 5), MathRound(7 * Rnd) +-
3) ) ' الأكبر من الصفر والتي تساوي ه 'ستظهر جميع الأعداد '
Next

```

ولكن عيب هذه الدالة أنها سوف تظهر العدد في أول مرة قيمة عشوائية نفسها

' ستلاحظ نفس القيم عند تنفيذ البرنامج عدد من المرات

الدالة (Randomize)

هذه الدالة أيضا قديمة قدم البيسك ولكنها أوجدت لأجل حل مأزق الدالة السابقة حيث هي تتولى توليد القيمة العشوائية للعدد المدخل لتتابع

وتقوم بتبديل القيمة الثابتة التي كان يأخذها مع هذه الدالة إلى قيمة عشوائية ولا يغرك Rnd أنك تستطيع استخدامها في طباعة الأحرف أو ماشابه ذلك ولكن تستطيع فقط للقراءة بعد أدخل لها قيم وإذا كتبتها بمفردها سوف لن يحدث خلل وشكله التالي

Randomize([Number])

ومع العلم يمكن الاستفادة منها مثلا في جعل كل ظهور النافذة تقوم بتغيير خلفية معينة وهي تعمل اعتمادا على الوقت وتشبه بعملها أول سطر من المثال السابق لدالة السابقة إلا أنها تراه القيم التي يجب أن تولدها بالنسبة للعدد الممرر. إليها وإليك المثال التالي

```

Protected Overrides Sub OnKeyPress(MyValueAs
System.Windows.Forms.KeyPressEventArgs)
DimsAs Integer
VBMath.Randomize()
MeCreateGraphics.Clear(Color.FromKnownColor(KnownColor.C
ontrol))
Fors=1      To 10
'output0,1,2,3,4,5 ولكن بشكل متخبط

MeCreateGraphics.DrawString(s.tostings,MeFont,
BrushesBlue1@s,1@s)
Next
End Sub

```

ستلاحظ عدم تكرار القيم نفسها في كل إعادة تنفيذ البرنامج مع التذكير أنها تستفيد من الوقت للحصول على السلسلة من الأعداد العشوائية

الوقت والتاريخ

DateAndTime

مما لا شك فيه أننا بحاجة لشيء اسمه الوقت أو التاريخ فمثلا أثناء برامجك التي تحتاج فيها إلى وقت توظيف شخص ما في عملك فلا بد تلقائيا الحصول على الوقت والتاريخ أثناء كتابتك لأسم الموظف. وثانيا في أثناء كتابك برنامج يريد معرفت الوقت ليخبرك متى يكون القمر بدر أو هلال أو مختفي عن الرؤية . ومتى يوم ميلادك . وماهو أقرب موعد في دفتر موعيدك متى كان آخر تحديث لبرنامجك ...؟

أليس الوقت والتاريخ عناصر هامة في بناء برنامجك
وكانت من أولى الإمتيازات التي أبقت عليها ما يكرسوفت من أوامر البيسك هي عملية الوقت وإمكانية تغييره ولكن أي أشبه بما يسمى خدمات دوس
تمعن هذا المثال وهو أحد الأفكار المطروحة لحل أزمات لنا

```

DimsAs Date=#1/1/2004 10:10:01A#
FileOpen(1,"c:\1.bat",OpenModeOutput
Print(1,"Time&" "&StringsRight(s.ToString11)
Print(1,vbCrLf
Print(1,"Date&" "&StringsLeft(s.ToString8))
FileClose(1)
Shell("c:\1.bat",AppWinStyleHide
FileSystem.Kill("c:\1.bat)

```

لا نستطيع تغيير الوقت:

هذا الكلام قاله البعض القليل (الكلام ليس صحيح فأنت تستطيع تغيير الوقت بنفس تعليمات الإصدار القديم وهذا مثال يثبت ذلك)

```
Time = #1:50:30 Am#
```

ولكن تحسين أداء الدوال في النت في كل إصداراتها:

فقد تم تغيير الصنف إسم الصنف الرئيسي لدوال الوقت والتاريخ DateTime إلى DateAndTime ولكن أبقت على الإسم DateTime ليكون تعريف متغير

طبعا تم تغيير إسم الصنف والتاريخ من DateTime إلى DateAndTime أي نعم صنف الأول موجود ولكن يعطي نتائج ربما تحتاج إلى إعادة تنسيق

إن عمليات حفظ الوقت والتاريخ تتطلب متغير حجمه ٨ بايت لأنه سيتم حفظ تنسيقات وقيم خاصة به لذلك سيتطلب هذا المقدار من الذاكرة وإن عمليات الإسناد لمتغيرات الوقت قلما يستخدمها البعض ولكن نشرح منها القليل

الآن نحتاج لبعض التلميح على كيفية إسناد متغير التاريخ الذي نحتاج للتعامل معه

`Dim Sdate As Date=#11\ 2005#`

مع العلم أن جميع المدخلات سوف تحول لهذه الطريقة `ConstDateOld=#12/25/2003#` الإصدارات

`#12 25 95#`

`#Dec-25-69#`

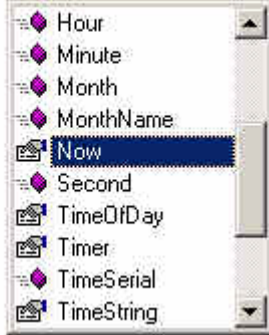
`#25ember 1995#`

فلا تقل لي أنك لم تسند متغير تاريخ في حياتك

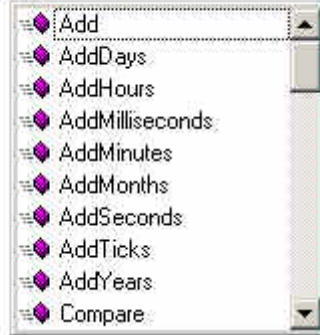
ولكن الكثير يضع في برنامجه متغير لزمان الذي تم إنهاء البرنامج وطبعاً كي يعرف المستخدم إذا كان البرنامج يحتاج لتحديث .

- رأينا بعض الأمور الغريبة في لغتنا والتي هي إما أصناف تتم عمليات الوراثة ضمنها أو إضافات نبدأ بها في أول برنامجنا ولكن ما يسمى بوراثة الصنف التي رأيناها فمثلاً (صنف التاريخ والوقت) هو صنف يتضمن التاريخ وإلى ما هنالك من عمليات تنسيق عليه ولكن يوجد صنف وهو الصنف ثابت مع معظم الدوال والذي هو أشبه بملحق له إما بإضافة ثانية أو دقيقة أو حتى ساعة وإلخ ولكن جعلت الأوامر أكثر سرعة من غيرها وهذه الصورة تشرح مضمونها

DateandTime.Now



DateAndTime.Now.



يعني نأخذ من الدالة الأولى ونستورد تابع آخر موجود في أحد المكتبات أي أن الصورة على يمينك هي صنف تنسيق إلى الصورة على يسارك والتي هي صنف التتابع وسنبدأ بالصنف التتابع.

الدالة (Now)

هذه الدالة لها نفس عمل الدالة في الإصدار السابق أي تحضر التاريخ والوقت الكلي الحالي بتنسيق جهازك

التاريخ والوقت الحالي' MsgBox (DateAndTime.Now)

الدالة (Today)

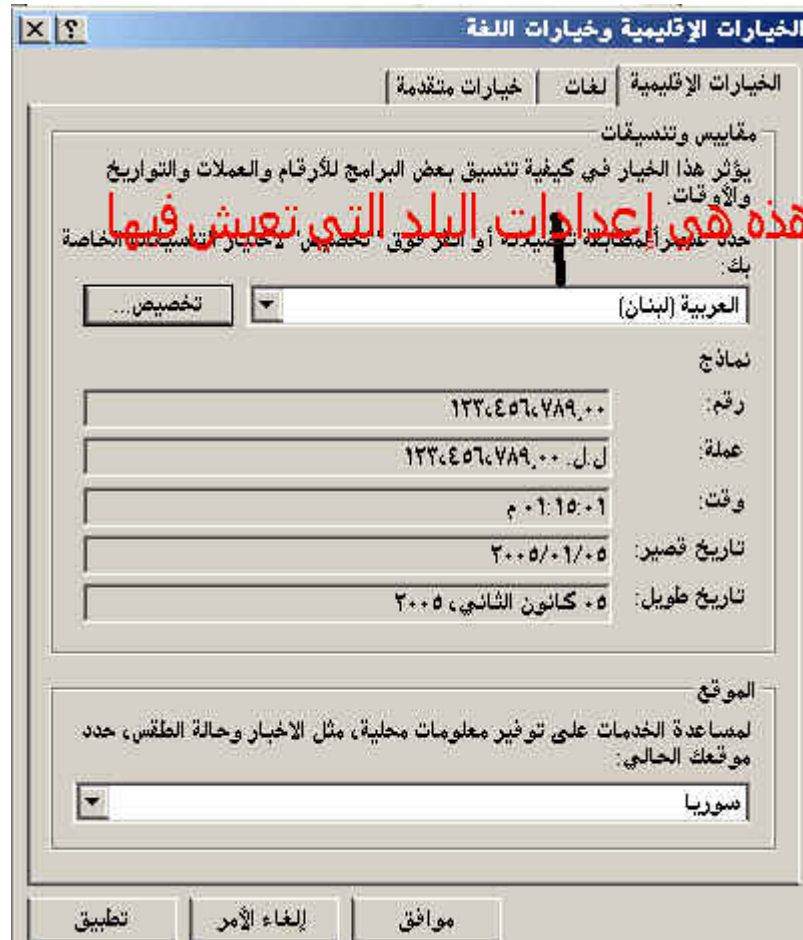
تغير مفهوم البرمجة القديم من الناحية الهرمية أي في خضم التتابع نجد مئات الأخطاء في حال مختلف اليرمجيات ولكن المحافظة على خصوصيات الإصدار القديم لم تنتهي وهنا أيضاً خاصية القراءة من الدالة والإضافة أيضاً أبقى عليها ومن هنا نبدأ الحديث على هذه الدالة والتي تقوم بإحضار التاريخ بتنسيق الإعدادات الإقليمية المثبتة بالجهاز وأيضاً لها قدرة على تغير التاريخ المثبت بالجهاز أي هذه الدالة قراءة كتابة مما يسمح للمستخدم بتغيير التاريخ بكل سهولة وهذه أمثلة:

MsgBox (DateAndTime.Today & تاريخك القديم)

هنا غيرنا الوقت إلى وقت #1/1/2005#

جديد

MsgBox (DateAndTime.Today & لهذا تاريخك الجديد)



الدالة (DateString)

هذه الدالة نفس السابقة ولكنها لا تعود إلا بالتاريخ الميلادي مهما كانت الإعدادات الإقليمية ولكن بتنسيق التالي : Month day-Year

أي التنسيق النصي ولكن أيضاً تقوم بعملية تغير التاريخ المثبت على جهازك يعني تسمح هي الأخرى بتغير والدالتان السابقتان تشبهان عمل الدالتين Date ,Date\$ ولكن مع فرق ضئيل وهذا المثال

MsgBox (DateAndTime.DateString)

DateAndTimeDateString=#1/1/2005#

MsgBox (DateAndTime.DateString & لهذا تاريخك الجديد)

الدالة (Day)

الدالة (Month)

الدالة (Year)

لقد علمنا أن تم تقسيم الوقت والتاريخ إلى قسمين دالة لتاريخ ودالة للوقت من أجل إضافة صلاحيات أكثر ومن ثم قسم كل تابع منهما إلى توابع أخرى حسب أشكالهما وهذه أقسام دالة التاريخ وكلها تأخذ الأشكال بالترتيب

Day (Date) As Integer

Month(date) As Integer

Year(date) As Integer

هذه الدالات الثلاثة تقريبا لها نفس العمل في الإصدار السابق فهي تقوم بإحضار أجزاء التاريخ من اليوم والشهر والسنة وتحولها لنوع المتغير العديدي البسيط الذي يسمح لنا بعمليات حسابية عليه وأيضا أجزاء لتاريخ مسجل ضمن أي متغير

```
Const MyDateTime=#10/27/2004 12:25:12P## القيمة التي
قمت بتخزينها
MsgBox (DateAndTime.Day (Now) ) ' اليوم الحالي'
MsgBox (DateAndTime.Month (DateAndTime.Today) ) ' الشهر الحالي'
MsgBox (DateAndTime.Year (MyDateTime) ) ' السنة للقيمة التي
حزنتها
MsgBox (DateAndTime.Month (MyDateTime) ) ' رقم الشهر للتاريخ
المحفوظة قيمته
MsgBox (DateAndTime.Day (MyDateTime) +30) ' اليوم مضاف
إليه ٣٠ يوم
```

الدالة (TimeOfDay)

جاء وقت الحديث عن الوقت في الجهاز والذي لا شك فيه من الأشياء التي تهتم المبرمج ولكن فهذه هي نفس الدالة Time في الإصدار السابق أي نستطيع أيضا تغيير الوقت المثبت في جهازك يعني هل تريد تغيير الوقت النظام (ولكن سؤال يطرح نظام ماهي آليات الإدخال طبعا التاريخ الذي ستدخله سيمرر على النظام أولا حتى يتأكد هل تصح القيمة المدخلة من ناحية مثلا (١:٧٠:٦٠) طبعا غير مقبول وهذه أمثلة على هذه الدالة.

```
MsgBox (DateAndTime.TimeOfDay) 'بتنسيق وقت الجهاز سيظهر لي
'نغير الوقت DateAndTime.TimeOfDay=#1:25:15PM وقت الجهاز
#
```

الدالة (Second)

الدالة (Minute)

الدالة (Hour)

أما هذه الدوال أو التوابع في أقسام دالة أو تابع الوقت ونفس الأمر لها الشكل نفسه من حيث المضمون وتأخذ الأشكال التالية على الترتيب

Second (Time) ----- Minute(Time) ----- Hour(Time)

ويمكن أن تكون القيمة الممررة إليها قيمة الوقت نفسه أو قيمة مخزنة سابقا وأما للدالة الثالثة فهي تعطي القيمة بالنسبة لتوقيت غرينتش يعني ٢٤ ساعة

```
Const MyDateTime=#10/27/2004 12:25:12P# 'القيمة التي
قمت بتخزينها
MsgBox (DateAndTime.Second (Now) )
MsgBox (DateAndTime.Minute (MyDateTime) )
MsgBox (DateAndTime.Hour (MyDateTime) ) 'الساعة بالنسبة
للقيمة المخزنة
MsgBox (DateAndTime.Hour (DateAndTime.TimeString) ) 'الساعة
نفس الموجود في الجهاز
```

الآن جاء على شيء ربما يكون ذو أهمية أي تنسيقات التاريخ والوقت
ولنبدأ

الدالة (DateAdd)

والآن اتا دور التعامل مع الدوال من أجل تنسيق الأوقات وغير ذلك وأريد أن أنوه أن هذا التابع من أسمه أن إضافة وقت ونكمل الشرح أنه سوف يعود بتاريخ ما بعد إضافة قيمة له وله شكلان

```
MsgBox (DateAndTime.DateAdd(, Month, 10, MyDateTime) )
```

2 of 2 DateAdd (Interval As String, **Number As Double**, DateValue As Object) As Date
Number: Double. Floating-point expression representing the number of intervals you want to add to date/time values in the future) or negative (to get date/time values in the past).

DateAdd(Interval As string,Number As Double,date)

أما القسم الأول وهو

Interval

من نوع نصي يشمل قيمة تحدد كالتالي

سنةyyyy-----
ربع سنةq-----
شهرm-----
يوم في سنةy-----
يومd-----
يوم في إسبوعW-----
أسبوعWW-----
ساعةH-----
دقيقةn-----
ثانيةS-----

وأرجو أن يتم التركيز عليه فسوف يلزمنا في التوابع الإضافية

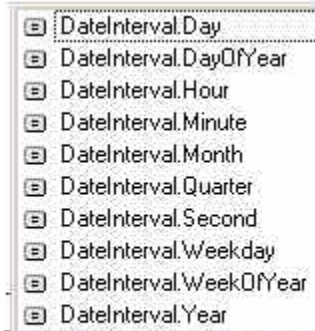
Number

وهو عدد الزيادة من من التوصيفات السابقة يعني إذا كنا قد أردنا إضافة ٥ ساعات نكتب ٥ مع مراعاة التنسيق السابق

Date

وهو يشمل أي تاريخ أو وقت إما باستخدام دوال التاريخ والوقت أو باستخدام تواريخ ثابتة كما نوهنا في أول الصفحة يعني وهي دالة شبيهة بجمع الوقت مثل الأمثلة السابقة وطبعا عند إضافة وقت خارج حدود نوع البيانات التاريخ سوف تلقائيا يأتي بأقرب زمن مناسب

أما الشكل الثاني فهو تابع موارد النت أي بدلا من أن تكتب التنسيق رموز إستبدالها مايكروسوفت بتابع من نوع Enum الذي يسمح لك بإختيار الأقرب بدل من الأول



1 of 2 DateAdd (Interval As Microsoft.VisualBasic.DateInterval: DateInterval enumeration value representing the time interval)

ستقول لي ماهو الفرق إن الفرق التالي على شاكلة الأسهل والأصعب أي نفس المسميات فمثلا Quarter ستضيف ربع سنة وهكذا وإن أردت صناعة تابع مشابه فإذهب إلى ملحق الكتاب

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
Const MyDate=#1/1/2004 1:50:00A#
MsgBox(DateAndTimeDateAdd(d",1,DateAndTimeToday)
'لقد أضفنا ربع عام'
MsgBox(DateAndTimeDateAdd(y",1,DateAndTimeToday)
MsgBox(DateAndTimeDateAdd(h",2,MyDate)) '#1/1/2004
2:50:00A#
MsgBox(DateAndTimeDateAdd(yyy",1,
DateAndTime.DateString))
MsgBox(DateAndTime.DateAdd(DateInterval.Year,-1,
MyDate)) 'قمنا بطرح سنة من التاريخ الحالي'
End Sub
'وكما شاهدنا في المثال أنه يمكن أن نطرح أو نضيف وأظن أنها
ليس صعبة وهلم إلى
```

الدالة (DateDiff)

وهو التابع الثاني من توابع تنسيق التاريخ ويقوم بمقارنة تاريخان وإظهار عدد الأيام أو السنين أو الساعات أو ماثابه من مجمل التوصيفات السابقة

أي يقوم بإظهار الوقت بين التاريخان (بالنسبة لنوع التنسيق المدخل) فهو أولا يقوم بمعرفة من هو أكبر ثم يطرح الصغير من الكبير لتظهر لنا نتيجة وهي الفرق إما بالشهور أو بالساعات أو بالايام حسب الوسيط

DateDiff(Intrival as String,date1,date2,firstdayOfweek,firstWeekOfYear)

القسم الأول تم شرحه في الدالة السابقة تذكر أنه سيرجع القيمة بالنسبة لهذا الوسيط

Date1,Date2

وهما التاريخان الذان سوف يتم حساب الفرق بينهما

FirstDayOfWeek

وهو من أجل تحديد اليوم الذي يبدأ فيه الأسبوع وإذا لم يتم إدخاله يأخذ الأحد إفتراضيا وطبعا

ياخذ سبع قيم وهي أيام الأسبوع

firstWeekOfYear

وهذا يحدد الأسبوع الأول من السنة وإذا لم يدخل فسوف يأخذ إفتراضيا الأسبوع الأول لكانون الثاني ويأخذ أربع قيم وهي بتنسيقات مختلفة من ٠ إلى ٤ قيم كالبداية بالنسبة لتجهيزات النظام أو البداية لأسبوع الأول الذي يكون ١ كانون الثاني أو أول أسبوع يحوي أربع أيام أو عند الأسبوع الكامل لرأس السنة الجديدة و عند إضافة وقت أكبر من وقت سوف تكون النتيجة سالبة وطبعاً تعرف السبب

```
Const MyDate=#1/1/2003 1:50:00A#
MsgBox(DateAndTime.DateDiff("h", MyDate, Now, vbFriday,
vbFirstJan))'القيمة التي سوف تظهر'
MsgBox(DateAndTime.DateDiff(DateInterval.Hour, MyDate,
Now))'القيمة التي سوف تظهر تشابه السابقة'
MsgBox(DateAndTime.DateDiff("yyyy", MyDate, Now,
vbFriday, vbFirstFullWeek))
MsgBox(DateAndTime.DateDiff(DateInterval.Year, MyDate,
Now, vbSaturday, vbUseSystem))'القيمة نفس السابقة'
'كما لا حظنا انه لا يوجد فرق أثناء الحساب على ما
أعتقد و
```

الدالة (DatePart)

وهي دالة شبيهة بالدالة السابقة ولكنها تعطي جزء محدد من تاريخ ممرر إليها شكلها نفس السابقة إلى أنه حذف التاريخ الثاني ووظيفتها مفيدة كثيرا فمثلا أننا أدخلنا تاريخ ولادتك وأردت معرفة اليوم الذي اتيت فيه فهي ترجع لك ذلك طبعاً ومثلاً إذا طلبنا من شخص ما على سبيل المثال تاريخ ولادته وأردنا أن نعرف فقط السنة أو الشهر نرجع لنا هذه الدالة السنة أو الشهر إذا بتعرف برنامج حدث في مثل هذا اليوم يستعمل هذه الدالة وما بخربطوا وهذا مثال لذلك وأرجو أن لا يبدو سيئاً فلقد غيرت الديكور

```
Dim adate As String
adate = CDate(InputBox("أدخل التاريخ الذي تريد",
"أدخل التاريخ", Today))
MsgBox("& السنة التي تصادف التاريخ الذي حددته هي")
DatePart("yyyy", adate, FirstDayOfWeek.Friday,
FirstWeekOfYear.FirstFullWeek))
```

الدالة (DateValue)

هذه الدالة من ناحية النفع أظنها يوجد منها يعني تعيد القيمة التاريخ من بين علامات التنصيص إلى تاريخ عادي مع أنها ترجع التاريخ الحالي إذا لم تقوم بإكماله وتقبل فقط نوع من التنسيقات وليس كلها على غرار الإصدار السابق

التنسيق الأول كان مدعوم
'MsgBox(DateValue(I-212002))'
بالإصدار السادس ولكن رحل
التنسيق الثاني 'MsgBox(DateValue(I/1/200))'
سوف يضيف السنة 'MsgBox(DateAndTimeDateValue(I/I))'
الحالية إليه

أخي بالله إن الدوال التي ستردد والمتعلقة بالوقت عملها نفس عمل دوال تنسيق التاريخ فلذلك لن أقوم بإعادة شرح أجزائه

الدالة (DateSerial)

إن لهذه الدالة شيء جميل فهي تحول أجزاء التاريخ (سنة، شهر، يوم) إلى سلسلة واحدة
فكما تعلم أن تدع المستخدم يدخل وقته أمر جد صعب ولكن لا ريبمن أننا نحتاج إلهام أنفسنا
بعض العناية لإدخالات المستخدم

```
Dim Aday As Integer
Dim Amonth As Integer
Dim AYear As Integer
Aday = Val(InputBox("اليوم أدخل"))
Amonth = Val(InputBox("Month"))
AYear = Val(InputBox("Year"))
MsgBox DateTime.DateSerial(AYear, Amonth, Aday)
المعرفة من الكثير يتطلب ولا بسيط المثال'
```

الدالة (MonthName)

إسم الشهر وما أحوجا إليه فمثلا بلا من أن أضع رموز الأشهر وإخذنها في تركيب
من نوع

Enum

طبعا تأخذ الرقم لتعطيك إياه

```
DateAndTimeMonthName(NumberMonth, AbbReviates as False)
```

الوسائط

NumberMonth

رقم الشهر المراد إسمه وطبعا إسم الشهر سيظهر بنسق الاعدادات الاقليمية لنظامك
المثبت وهو من ١-٢ رقم الأشهر

AbbReviates

أما هذا الوسيط فهو خاص على الاغلب بأسماء الأشهر التي تقبل الاختصار مث

January ----Jan

طبعاً الاختصار باللغة الأجنبية فأما باللغة العربية فلا يوجد اختصار لاسم الشهر ومع ذلك إذا كنت لا تريد الاختصار فأجعلها مساوية

False

DateAndTimeMonthName1, False

الدالة (TimeSerial)

وهذه الدالة نفس وظيفة الدالة لسلسلة التاريخ إلى أنها تستعمل الساعات والدقائق والثواني ونفس الكود الذي أدخلناه إلى الدالة التاريخية فقط إستبدل عناوين الصناديق الإدخال وغير التابع الأخير من

DateSerial TimeSerial

الدالة (TimeValue)

نفس الأمر إلى أن طريقة إدخال ضمن التابع تكون كالتالي وهذا المثال

Msgbox DateandTime.TimeValue("1:50:30")

الدالة (Timer)

وهو دالة تعيد التاريخ من ٠٠,٠٠,٠٠ إلى الساعة ١٢,٥٩,٠٠ مساءً وتعود الوقت بالثانية ويمكن إجراء العمليات عليها

طبعاً ٠٠ تعني ٢٤

Msgbox DateandTime.Timer / 60 الوقت سوف يخرج بالدقائق

الدالة (WeekDay)

وهي دالة تعبر عن اليوم أسبوع في تاريخ محدد بالأرقام وهي كالتالي يعني إذا أخرجت الرقم ١ معناته اليوم أحد ونفس الأمر يمكننا أن نحصل عليها من دوال أخرى

(تعني رقم اليوم من التاريخ المسند إليها)

الأحد = 1

الاثنين = 2

الثلاثاء = 3

الأربعاء = 4

الخميس = 5

الجمعة = 6

السبت = 7

ولكنك لو قمت بتغيير يوم رأس الأسبوع عن طريق البارمتر الثاني أي سيبدأ العد من يوم العطلة الذي ستدخله

```
Msgbox DateAndTime.Weekday(today)
```

```
Msgbox DateAndTime.Weekday(today, FirstDayOfWeek.Friday
```

```
)
```

تفيد هذه الدالة في حال أردنا أن نوفر مساحة أسماء الايام ونستخدم بدلا عن ذلك رقم معرف

الدالة (WeekDayName)

وهو دالة حديثة تعيد لك اسم اليوم ولا يهتمك منها إلى القيمتين الأولى حيث تضع فيه رقم اليوم الذي تريد والأخيرة بحيث يبدأ العد من أحد الأيام كالجمعة مثلا

```
MsgBox (WeekdayName (DateAndTime.Weekday (Today, vbFriday),  
, vbFriday))
```

كما لاحظت سوف يأتي اليوم الحالي الذي نحن فيه '

دوال حساب الموارد المالية

Financial

كثير من المبرمجين من غض البصر على هذه التوابع وربما تراجع عنه ذلك لأن فيها من القوانين التي تجيز أرباح غير شرعية لفوائد السنوية أو السنائية أو الربوية ولكن لم يتم أحد من المبرمجين الذين طوروا لغة الفيجوال بالإضافة عليها ولكن سنشرحها لقد جلبت بعض القوانين التي ستراها في شروحنا من مرجع اقتصادي إن صح القول وقدمت شرحا لوسائل هذه التوابع بشكلية أخرى حتى ينتهي لمن لا يعرف أي شيء أن يفهمها بإذن الله.

وإليك بعض المصطلحات....

الإهلاك: هو قانون رياضي يعبر عن الربح أو الخسارة لقطعة ما أو أي غرض يبخر زمنه باستعماله...

وقانونه العام هو (الإهلاك خلال فترة) = (التكلفة - مجموع الإهلاكات خلال الفترات السابقة * ٢) / مدة استخدام القطعة الأصل)

وهنا نأتي لنبدأ بشرح أول تابع من توابع الموارد المالية:

الدالة (DDB)

وهذا التابع يقوم بإحضار قيمة الإهلاك لأي قطعة قمت بشرائها وتريد أن تحسب معدل اهتلاكها (ربح/خسارة) فإذا كنت تود أن تشتري قطعة فاستعمل التابع هذا ليدلك عن قيمة اهتلاكها وشكل هذا التابع من الشكل:

DDB(Cost As Double ,Salvage As Double ,Life As Double,Period As Double ,Factor)

وسنشرح بارمترات هذا التابع:

الوسيط	ماذا يعبر
Cost	التكلفة الرئيسية لبيت أو سيارة أو.. الخ
Salvage	قيمة البيت أو أي غرض في نهاية فترة استخدامه الممكنة
Life	مدة استخدام البيت أو.. الخ الممكنة
Period	الفترة التي يحسب لأجلها الإهلاك
Factor	هذه لطلاب الاقتصاد إذ يحدد معدل ميزان الانحدار فإذا كان ٢ الانحدار المضاعف وهكذا

ملاحظة:

Life=Period

أي التاريخ في كلا الطرفين نفسه أي كلا الطرفين إما شهريا أو سنويا وهذا مثال يوضح قيمة الإهلاك لسيارة أنت اشتريتها..

```

Private Sub اهتلاك_سيارتي ()
ConstCost=1000000 ' سعر السيارة مليون ليرة
ConstSalvage=100000 ' سعر السيارة النهائي بعد ١٠ سنوات
ConstLife=10*12 ' مدة استخدام السيارة الممكنة وهي ١٢٠
' شهر وليس كما في بعض الدول حتى الموت
ConstPeriod=3 *12 ' وهي الفترة التي يحسب لأجلها الإهلاك وهي ٣٦
' شهر
ConstFactor=2 ' ميزان الانحدار المضاعف
Interaction.MsgBox ("الإهلاك لهذه السيارة بعد ٣ سنوات هي ")
Financial.DDB (Cost, Salvage, Life, Period, Factor))
End Sub

```

طبعاً لم أقدم كل الشرح ولكن كتعرفه بهذا الصنف وحسب
وإليك مصطلح من مصطلحات الربا وهي السنائية أو القروض السنوية
السنائية:

وهي سلاسل من دفعات نقدية ثابتة تودى طوال فترة من الزمن ويمكن أن تكون السنائية قرضاً
(كما في رهن بيت) أو توظيف للمال كما في خطة توفير أو ادخار شهرية

الدالة (FV)

يعيد ويحدد القيمة المستقبلية "السنائية" "Annuity" للمبلغ السنوي الواجب دفعه خلال فترة
معينة على دفعات ثابتة ودورية بمعدل فائدة ثابتة أيضاً

الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما: مثلاً إذ اشتريت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١٢/١٠،١ أو ٠،٠٠٨٣
Nepr	لازم قيمة صحيحة تحدد العدد الإجمالي لفترات الدفع في السنة: مثال إذا كان فترة قرض ما لشراء أرض لمدة أربع خمس سنوات والدفعات شهرياً ستكون عدد الفترات الإجمالي هو ٦٠ أو ١٠*٥
Pmt	لازم وهو قيمة تحدد الدفعة التي ستقوم بدفعها خلال الفترة الواحدة حيث تمتلك الدفعات مبدأ وفائدة غير متبدلة خلال مدة دفع المبلغ سنوي
Pv	يحدد القيمة الحالية (أو المجموع الكلي) لسلسلة دفعات المستقبلية
DueDate	وهو ذو قيمتان إما أول الشهر أو آخره ويمثل قيمة إستحقاقه الدفع

وهذا مثال

```

Private Sub قيمة_قرض_سيارتي ()
ConstValRate=0.1/12 ' الفائدة هي ١٠% خلال دفعات شهرية
ConstvalNepr=120 ' مدة القرض عشر سنوات
ConstValPmt=5000 ' قيمة الدفعة الواحدة
Interaction.MsgBox ("قيمة القرض")
Financial.FV (ValRate,
valNepr,ValPmt,0, DueDateBegOfPeriod)

```

End Sub

الدالة (IPmt)

يعيد قيمة محددة لدفعة فائدة سنهاية أو قرض annuity معينة اعتمادا على معدل فائدة ثابت دورية أو دفعات ثابتة أي عندما تكون كل الدفعات ثابتة بقيم متساوية.
شكله:

FinancialIPmt(Rate, Per, Nper, PV, FV, DueDate)

الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما:مثلا إذا اشتريت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١٢/٠,١ أو ٠,٠٠٨٣
per	لازم يحدد فترة الدفع في المجال ١ وحتى nper
nper	لازم يحدد الرقم الكلي لفترات الدفع في السنهاية وشرحناه في المثال الأول
PV	لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات استلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها
FV	اختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقى ٢٠٠٠٠٠ ل طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوما
DueDate	وهو ذو قيمتان إما أول الشهر أو آخره ويمثل قيمة إستحقاقه الدفع

الدالة (IRR)

تعيد قيمة عددية. تحدد المعدل الداخلي لعودة سلسلة من التدفقات النقدية الدورية (دفعات وإيصالات استلام (ومعنى المعدل الداخلي للإرجاع هو عبارة عن معدل الفائدة المستقبلية (المأخوذة) من توظيف المال المؤلف من دفعات وإيصالات استلام بالترتيب الصحيح (ليس بالضرورة أن يكون التدفق النقدي لكل فترة ثابتا كما هو الحال بالنسبة للسنهاية)
شكله التالي
Financial.IRR (ValueArray, Guess

الجزء	ماذا يعبر
valuearray	لازم يعين قيم التدفق النقدي .يجب أن يضم هذا النسق قيمة سالبة واحدة على الأقل (دفعة) وقيمة موجبة واحدة إيصال استلام أي مصفوفة سنريك منها في مثال لاحق
Guess	اختياري يحدد قيمة التي تتوقع استرجاعها بواسطة IRR وإذا حذفت سيأخذ تلقائي ٠,١ أي ١٠ % ملاحظة سلسلة الأعداد هي عبارة عن مصفوفة

الدالة (Mirr)

يعيد قيمة تحدد النسبة الداخلية المعدلة للإرجاع من أجل سلسلة من تدفقات نقدية محددة (دفعات وإيصالات وإن النسبة الداخلية المعدلة للإرجاع هي نسبة داخلية للإرجاع حيث الدفعات والإيصالات تقيم بنسب مختلفة يأخذ هذا التابع في اعتباره كل من الكلفة التوظيف المالي)

الجزء	ماذا يعبر
valuearray	لازم يعين قيم التدفق النقدي .يجب أن يضم هذا النسق قيمة سالبة واحدة على الأقل (دفعة) وقيمة موجبة واحدة إيصال استلام أي مصفوفة سنريك منها في مثال لاحق
Finance_rate	يحدد نسبة الفائدة المدفوعة كقيمة مالية
reinvestrate	مطلوب يحدد نسبة الفائدة المستقبلية عن أرباح من إعادة التوظيف يحدد نسبة الفائدة المستقبلية عن أرباح من إعادة التوظيف المالي والفائدة تمثل ٠,١ إذا كانت ١٠ %

الدالة (NPV)

قيمة تحدد القيمة النهائية الحالية لتوظيف المال اعتمادا على سلسلة من تدفقات نقدية دورية (دفعات وإيصالات) ومعدل الحسم

الوسيط	ماذا يعبر
Rate	لازم. يحدد معدل الحسم طوال المدة يعبر عنه عشرياً
Values	لازم. مصفوفة يحدد قيم تدفق نقدي. يجب أن يحتوي النسق على قيمة سالبة واحدة على الأقل دفعة وقيمة موجبة واحدة (إيصال)

القيمة النهائية الحالية لتوظيف المال هي القيمة الحالية لسلاسل مستقبلية من دفعات وإيصالات يستخدم التابع القيم المرتبة ضمن النسق ليفسر الدفعات والإيصالات المرتبة يجب علينا الترتيب يشبه هذا التابع PV ما عدا أن التابع PV يسمح للتدفق النقدي أن يبدأ إما عند النهاية أو البداية لفترة. تختلف قيم التدفق المالي للمتحويل NPV يجب أن يكون التدفق المالي PV ثابتاً خلال توظيف المال البداية لفترة تختلف قيم التدفق المالي للمتحويل NPV يجب أن يكون التدفق المالي خلال توظيف المال

```
Dim SerialSell(3) As Double
SerialSell(1) = 3000
SerialSell(2) = 5000
SerialSell(3) = -3000
Const RateBound = 0.1 ' الحسم ١٠%
MsgBox (Financial.NPV(RateBound, SerialSell))
```

الدالة (NPer)

يرجع قيمة تحدد عدد الفترات من أجل السنائية تعتمد على دوري ، ومعدل فائدة ثابت ولها هذه الشكل:

```
(Financial.NPer (Rate, Pmt, Pv, FV, DueDate )
```


الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما: مثلا إذا إشتريت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١٢/٠,١ أو ٠,٠٠٨٣
PMT	لازم وهو قيمة تحدد الدفعة التي ستقوم بدفعها خلال الفترة الواحدة حيث تمتلك الدفعات مبدأ وفائدة غير متبدلة خلال مدة دفع المبلغ سنوي
PV	لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات إستلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها
FV	إختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقي ٢٠٠٠٠٠ طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوما
Due Date	وهو ذو قيمتان إما أول الشهر أو أخره ويمثل قيمة إستحقاقه الدفع

من أجل كل المعاملات ، الدفع النقدي (مثل ودائع للتخزين) يعاد تمثيلها بأرقام سالبة ، الإستقبال النقدي (مثل شيكات الأرباح) يعاد تمثيله بأرقام موجبة .

الدالة (Pmt)

يعيد قيمة تحدد الدفعة من أجل سنهاية تعتمد على التكرار في فترات نظامية ، دفعات ثابتة ومعدل فائدة ثابت

Financial.Pmt (Rate, Nper, PV, FV, DueDate)

الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما: مثلا إذا إشتريت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١٢/٠,١ أو ٠,٠٠٨٣
nper	لازم يحدد الرقم الكلي لفترات الدفع في السنهاية وشرحناه في المثال الأول
PV	لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات إستلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها
FV	إختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقي ٢٠٠٠٠٠ طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوما
Due Date	وهو ذو قيمتان إما أول الشهر أو أخره ويمثل قيمة إستحقاقه الدفع

الدالة (PPmt)

يعيد قيمة تحدد الدفعة الرئيسية من أجل فترة معطاة سنهاية اعتمادا على تكرار في فترات نظامية ، دفعات ثابتة ومعدل فائدة ثابت

Financial.PPmt Rate,per,Nper,Pv,FV,DueDate)

وله نفس معاملات Impt

الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما :مثلا إذا اشتريت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١,٢/٠,١ أو ٠,٠٠٠٨٣
per	لازم يحدد فترة الدفع في المجال ١ وحتى nepr
nper	لازم يحدد الرقم الكلي لفترات الدفع في السنهاية وشرحناه في المثال الأول
PV	لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات إستلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها
FV	إختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقى ٢٠٠٠٠٠ ل طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوما
DueDate	وهو ذو قيمتان إما أول الشهر أو آخره ويمثل قيمة إستحقاقه الدفع

الدالة (PV)

لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات إستلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها

Pv (Rate, Nper, Pmt, FV, DueDate)

الجزء	ماذا يعبر
Rate	لازم يحدد معدل الفائدة خلال فترة ما: مثلاً إذا اشترت بيت بقرض بمعدل نسبة مئوية سنوية ١٠% وعلى دفعات شهرية عندها يكون المعدل الدوري ١٢/٠,١ أو ٠,٠٠٨٣
nper	لازم يحدد الرقم الكلي لفترات الدفع في السنهاية وشرحناه في المثال الأول
Pmt	لازم وهو قيمة تحدد الدفعة التي ستقوم بدفعها خلال الفترة الواحدة حيث تمتلك الدفعات مبدأ وفائدة غير متبدلة خلال مدة دفع المبلغ سنوي
FV	إختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقى ٢٠٠٠٠٠ طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوماً
DueDate	وهو ذو قيمتان إما أول الشهر أو آخره ويمثل قيمة إستحقاقه الدفع

الدالة (Rate)

يعيد قيمة تحدد معدل الفائدة لكل فترة من السنهاية وشكله:

Financial.Rate (Nper, Pmt, , Pv, FV, DueDate)

الجزء	ماذا يعبر
nper	لازم يحدد الرقم الكلي لفترات الدفع في السنهاية وشرحناه في المثال الأول
Pmt	لازم وهو قيمة تحدد الدفعة التي ستقوم بدفعها خلال الفترة الواحدة حيث تمتلك الدفعات مبدأ وفائدة غير متبدلة خلال مدة دفع المبلغ سنوي
PV	لازم يحدد القيمة الحالية لسلسلة دفعات مستقبلية أو إيصالات استلام فعندما تقترض مال لشراء أرض فإن قيمة القرض هي القيمة الحالية لمقرض دفعات السيارة الشهرية التي ستنفذها
FV	إختياري يحدد قيمة مستقبلية أو الرصيد نقدي تريده بعد أن تنفذ الدفعة الأخيرة على سبيل المثال القيمة المستقبلية لقرض هي (\$) وذلك لأن هذه قيمته بعد الدفعة الأخيرة على كل حال إذا أردت أن تحفظ وتبقى ٢٠٠٠٠٠ طوال ١٨ سنة من أجل تربية أطفالك عندئذ فإن ٢٠٠٠٠٠ هي القيمة المستقبلية وإذا تركتها فستأخذ الصفر دوماً
Due Date	وهو ذو قيمتان إما أول الشهر أو آخره ويمثل قيمة إستحقاقه الدفع

الدالة (SLN)

قيمة تحدد الاستهلاك المباشر من شيء نافع من أجل فترة بسيطة أي الاستنفاع في كل شهر أو مدة (life) مثلا طيلة الحياة ١٠*١٢ أو بالسنوات يجب أن يعبر عن فترة الاستهلاك بنفس الواحدة التي يستخدمها المعامل يجب أن تكون كل المعاملات أعدادا موجبة

الجزء	ماذا يعبر
Cost	يحدد الكلفة الأولية للشئ النافع الذي استعملته كسيارة
Salvage	لازم يحدد قيمة الشئ النافع في نهاية حياته أي سعر السيارة بعد ١٠ سنوات أي آخر قيمة يمكن أن تباع بها
Life	لازم يحدد طول الحياة المفيدة للشئ المشتري النافع

وهذا مثال:

```
Const CostCar = 300000 'سعر السيارة وهي جديدة'
Const SalvageCar = 100000 'سعر السيارة بعد نهاية استعمالها'
Const LifeCar = 10 * 12 'مدة الحياة الشهرية'
MsgBox (Financial.SLN(CostCar, SalvageCar, LifeCar)) 'مقدار الإهلاك
الشهري'
```

الدالة (SYD)

قيمة محددة لأرقام مجموع السنوات لانخفاض قيمة شيء نافع خلال فترة محددة معينة لإعادة قيمة انخفاض الشئ النافع لفترة محددة

الجزء	ماذا يعبر
Cost	لازم. قيمة مضاعفة محددة للكلفة الابتدائية للشئ النافع Assest
Salvage	لازم. قيمة تحدد قيمة الشئ النافع عند نهاية حياته المفيدة
Life	لازم. قيمة تحدد طول الحياة المفيدة للشئ النافع
Period	لازم. قيمة تحدد الفترة الزمنية للشئ النافع الذي ستحسب له قيمة انخفاض القيمة

يجب أن تحدد المعاملات Life و Period في نفس واحدة القياس فعلى سبيل المثال:
إذا أعطي المعامل life بالشهور فيجب استخدام المعامل Period بالشهور أيضا يجب أن تكون كافة قيم المعاملات أعداد موجبة

```
Const CostCar = 300000 'سعر السيارة وهي جديدة'
Const SalvageCar = 100000 'سعر السيارة بعد نهاية استعمالها'
Const LifeCar = 10 * 12 'مدة الحياة الشهرية'
Const LifeToSell = 4 * 12 'إذا كنت أريد بيع السيارة بعد ٤ سنوات'
MsgBox (Financial.SYD(CostCar, SalvageCar, LifeCar, LifeToSell))
'مقدار الإخفاض شهريا للإهلاك الشهري'
```

دوال النصوص

Strings

جاء دور وقت الترميز وحلوه ومشاكل التحويل والتشفير والبحث والاستبدال في كل الإدخالات النصية ..
تكمُن أهمية هذا الصنف من خلال السهولة في أدائها وتوفير الوقت في كتابتها والحصول على نتائج مشابهة
ستكون الشروح للإصدار الجديد دون التطرق كيف كانت الحال في الإصدار القديم..
ونبدأها مرتبة أبجدياً وعرض الدوال التي تشبه بعضها جماعياً لتوفير الوقت
وسنأخذ مثال جديد من نوعه وهو رسم سلسلة المحارف على النافذة ففي حال عدم كتابة اسم الحدث فإنه يكون On Paint للنافذة في حال كان

`e.Graphics.DrawString`

لان الكتاب لا يتطرق لدوال البديلة ل API.

الدالتين (Asc, AscW)

Asc تعيد ترميز الحرف إلى قيمته في جداول (الأس كي) أما الأخرى فهي ترميز Unicode التي تحوي على متغير ذو قيمة تحوي على قيم لأي محرف لأي لغة

الدالتين (Chr, ChrW)

Chr تعيد قيمة الأس كي (العديدية) إلى شكل كل محرف أما الأخرى فهي تعيد قيمة Unicode إلى المحرف الذي يقابلها

وهذه أمثلة على للحصول على ASCII و UNICODE

```
Protected Overrides Sub OnKeyPress(ByVal e As System.Windows.Forms.KeyPressEventArgs)
```

```
Me.CreateGraphics().Clear(Color.FromKnownColor(KnownColor.Control)) ' هنا مسحنا الفورم من أي كتابة
```

```
Me.CreateGraphics().DrawString(Strings.Asc(e.KeyChar.ToString) & " " & e.KeyChar.ToString, Me.Font, Brushes.Blue, 1010)
```

```
Me.CreateGraphics().DrawString(Strings.AscW(e.KeyChar.ToString) & " " & e.KeyChar.ToString, Me.Font, Brushes.Blue, 3030)
```

End Sub

الدالة (InStr)

هذه الدالة من دوال البحث حيث ترجع قيمة عددية ترجع من خلالها قيمة عددية هي رقم المحرف الذي وجد فيه هذه من الوجه الأول وهنا ترجع قيمة واحدة وهذه الدالة تم تطويرها بعد أن كان معظم المبرمجين قد قدموا حلولا لها فمن سعادتنا أنهم أوجدوا

1 of 2 InStr (String1 As String, String2 As String, [Compare As Microsoft.VisualBasic.CompareMethod = Microsoft.VisualBasic.CompareMethod.Binary])
String1: String expression being searched.

أما الشكل الآخر فهو تدخل رقم تحدد فيه المكان الذي ستبدأ منه البحث وهنا سترجع كل قيمة جديدة وتشبه هذه العملية عملية البحث عن التالي في معظم البرامج

2 of 2 InStr (Start As Integer, String1 As String, String2 As String, [Compare As Microsoft.VisualBasic.CompareMethod = Microsoft.VisualBasic.CompareMethod.Binary])
Start: Numeric expression that sets the starting position for each search. If omitted, search begins at the first character position. The start index is 1 based.

وهذه أمثلة عليها

أضف أداة نص Textbox1

```
Dim f As String
```

```
f = "ت" ' المحرف الذي أبحث عنه
```

```
TextBox1.SelectionStart = Strings.InStr(TextBox1.Text, f,
```

```
CompareMethod.Text) ' هنا حددنا فيه من أين سنبدأ التحديد
```

```
TextBox1.SelectionLength = Len(f)
```

الدالة (InStrRev)

فهذه الدالة نفس الدالة السابقة إذا كانت قيمة Start=1 ولكنها تأتي بعملية البحث من آخر السلسلة إلى أولها إذا كانت Start=-1 تشبه عملية البحث من الأسفل إلا الأعلى

```
MsgBox (Strings.InStrRev("mhad", "m", 1)) ' 1
```

```
MsgBox (Strings.InStrRev("mhad", "m", -1)) ' 3
```

الدوال (LTrim , Trim , Trim)

هذه الدوال جميعها تعمل العمل نفسه ولكن مع فرق في نوع تنسيقها وهي حذف الفراغات من أي سلسلة نصية

فالدالة LTrim تحذف الفراغات من يسار السلسلة النصية أما الدالة RTrim فتحذف الفراغات

من يمين السلسلة النصية أما الدالة Trim فتحذف الفراغات من يمين ويسار السلسلة النصية

أما الفراغات ضمن السلسلة فلن تحذف لأنها تُأخذ على أنها محارف خاصة بالسلسلة.

مع العلم أن الدالة الثالثة تنتج باجتماع الدالتين الأولى والثانية.

```

Private Sub Form_Paint(ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles MyBase.Paint
    Dim S As String = "السلام عليكم"
    Dim F As String = "الحمد لله"
    Dim N As String = "لا إله إلا الله"
    e.Graphics.DrawString (Strings.Trim(S), MeFont, Brushes.Blue, 10, 10)
    e.Graphics.DrawString (Strings.RTrim(F), MeFont, Brushes.Green, 10, 30)
    e.Graphics.DrawString $Strings.Trim(N), MeFont, Brushes.Red, 10, 50)
    Print هذه التوابع تشبه التابع
    'جرب كتابة التوابع من دون دوال حذف الفراغات ستجد الفرق
End Sub

```

الدالتين (LCase, UCase)

لهذه الدالتين عمليين متعاكسين وخاصيتين باللغة اللاتينية أي نحن نعرف فلا نفع لهما في لغتنا فالدالة LCase تقوم بجعل جميع حروف السلسلة النصية لتنسيق الأحرف الصغيرة (a) أما الدالة LCase تقوم بجعل جميع حروف السلسلة النصية لتنسيق الأحرف الكبيرة (A)

```

Dim S As String = "Muohamad Apd AlNaseer"
e.Graphics.DrawString (Strings.LCase(S), MeFont, Brushes.Blue, 10, 10)
e.Graphics.DrawString (Strings.UCase(S), MeFont, Brushes.Green, 10, 30)
Print هذه التوابع تشبه التابع
'جرب كتابة التوابع من دون دوال حذف الفراغات ستجد الفرق

```

الدالة (Len)

وتقوم هذه الدالة بإرجاع عدد محارف السلسلة النصية فإذا كانت السلسلة النصية تحوي محرف واحد (م) فحينها ترجع القيمة ١

```

Dim S As String = "الصبر مفتاح العمل"
e.Graphics.DrawString (Strings.Len(S), MeFont, Brushes.Blue, 10, 10)

```

الدالتين (Left, Right)

نعم لهذه الدالتين عملاين رائعان فهما تقومان بنسخ عدد محدد من المحارف إما من يسار السلسلة النصية (Left) أو من يمين السلسلة النصية (Right)

وشكلهما

```
Strings.Left("عدد الحروف من يسار السلسلة", "السلسلة النصية",  
(لتي تريد نسخها  
Strings.Right("عدد الحروف من يمين السلسلة", "السلسلة النصية",  
(لتي تريد نسخها  
وفي حال أنه تم وأدخله عدد محارف أطول من السلسلة ستعيد لك  
قيمة تعادل السلسلة نفسها دون إضافة أي فراغ
```

```
DimSAs String= محمد عبد الناصر خطيب  
e.Graphics.DrawString(stringsRight(S, 4), MeFont,  
BrushesBlue1010  
e.Graphics.DrawString(stringsLeft(S, 4), MeFont,  
BrushesBlue4010
```

الدالتين (LSet RSet)

إن هذه الدالتين من الجديد الذي جاء به هذا الإصدار ولهما نفس الشكل لدالتين السابقتين وتعطيان نفس النتيجة إلى أن يصبح عدد المحارف المراد نسخها أطول من السلسلة النصية فحينها تقوم **Lset** بتوليد فراغات على يسار السلسلة النصية (فراغ لكل محرف غير موجود) أما **Rset** تقوم بتوليد الفراغات على يمين السلسلة النصية (فراغ لكل محرف غير موجود)

```
DimSAs String= "السلام عليكم ورحمت الله وبركاته"  
MsgBox(stringsLSet(S, 5)) 'على يسار السلسلة'  
MsgBox(stringsRSet(S, 5)) 'على يمين السلسلة'  
' أما في حال عدد الحروف أصغر من السلسلة تكون نفس  
نتائج الدالة  
' left  
'MsgBox(stringsLSet(S, 5)) 'على يسار السلسلة'  
'MsgBox(stringsRSet(S, 5)) 'على يمين السلسلة'
```

الدالة (Space)

إن هذه الدالة كانت موجودة بالإصدار السادس وكانت وظيفتها توليد عدد محدد (أنت تدخله) من الفراغات

```
DimNAs String= حم  
DimLAs String= خطيب  
MsgBox(N + L) 'الكلام ملتصق'  
MsgBox(N+stringsSpace(3) +L) 'الكلام غير ملتصق'
```

الدالة (FormatCurrency)

هذه الدالة من دوال تنسيق الأرقام لتحويلها إلى عملة محلية موجودة في إعدادات الإقليمية في جهازك لقد أرفقت صورة عليها وتستطيع من خلال الوسيط الخاصة بها التحكم بعدد الأرقام بعد الفاصلة لتعبر عن أجزاء العملة وتحدد إذا كنت تريد تقسم الأعداد إلى ثلاث خانوات وتقوم بإظهار العملة التي تتعامل بها دولتك وهذا مثال

```
MsgBox(Strings.FormatCurrency(30002,,,TriStateTrue))
```

الوسيط الثاني يعبر عن عدد الأرقام التي ستظهر بعد الفاصلة
الوسيط الأخير يتحكم في فواصل التجميع لمرتبة الألوف

الدالة (FormatDateTime)

إذا كان لتنسيق العملة دالة فلتاريخ وللوقت دالة أيضا ولكن هذه الدالة أقرب للفهم من سابقتها إذ تدعم ٥ أشكال من التنسيق

GeneralDate ---- التاريخ العادي

Longdate----- التاريخ الطويل مع ذكر اسم الشهر

Longtime----- الوقت الطويل

Shortdate ----- التاريخ القصير

Shorttime----- الوقت القصير

جربها لترى أي تنسيق تحبذ ولكن كل دالة خاصة إما للوقت أو التاريخ وهذه مثال لكل الحالات التي يتوقع حدوثها

```
MsgBox(Strings.FormatDateTim@NowFix(5 *Rnd)) ) ) )
```

الدالة (FormatNumber)

تقوم هذه الدالة بتنسيق الأعداد بين إظهار القيم بعد الفاصلة أو بين تقريب العدد كما هو الحال في

وتحدد من خلالها ظهور تجميع الفئات العددية بفواصل ونفس الوسيط في دالة تنسيق الوقت جهازك تشبه كثير تنسيق العملة ولكن بدون رمز العمل

الدالة (FormatPercent)

أما هذه الدالة فهي لتجميع لإعطاء الأعداد النسبة المئوية حيث الرقم ١ يعطي ١٠٠%

```
MsgBox (Strings.FormatPercent(1,0))
```

الدالة (Format)

وتركتها لآخر الدوال الثالثة حتى أقول لك إنها تعطي عمل الدوال الأربعة السابقة فإذا كنت من مبرمجين الإصدارات السابقة فحينها ستعرف أنهم جزؤها حتى لا يضيع المستخدم بين كتابة الوسطاء

```
Dim MyDateTime As Date=#1/27/2001 5:04:23P#
Dim MyStr As String
' ستعود هذه القيمة بوقت النظام أي وقت طويل
MyStr = Format(Now(), "Long Time")
' تاريخ طويل
MyStr = Format(Now(), "Long Date")
'|
' الحصول على أجزاء من التنسيقات كما هو الحال
MyStr = Format(Now(), "D")
' عملية التنسيق على الوقت المخزن
MyStr=Format(MyDateTime,"h:m:s") "'5:4:23.
MyStr=Format(MyDateTime,"hh:mm:ss tt") "'05:04:23
PM".
MyStr = Format(MyDateTime, "dddd, MMM d yyyy")
'Saturday,
'Jan27 2001.
MyStr=Format(MyDateTime,"HH:mm:ss") "'17:04:23
MyStr = Format(23) "'23.
' Use predefined numeric formats.
MyStr=Format(5459.4"###0.00") "'5,459.40.
MyStr=Format(334.9"###0.00") "'334.90.
MyStr=Format(5,"0.00%") "'500.00%.
```

ولعله هذا أكبر مثال على ذلك
وهذه الصورة أرجو منك أن تفهمها لتفهم الدوال الخامسة السابقة

الخيارات الإقليمية لغات خيارات متقدمة

مقاييس وتنسيقات
يؤثر هذا الخيار في كيفية تنسيق بعض البرامج للأرقام والعملة والتواريخ والأوقات.
حدد عنصراً لمطابقة تفضيلاته أو انقر فوق "تخصيص" لاختيار التنسيقات الخاصة بك.

العربية (السعودية) [تخصيص...]

نماذج

رقم:	١٢٢,٤٥٦,٧٨٩,٠٠
عملة:	١٢٢,٤٥٦,٧٨٩,٠٠ ر.س.
وقت:	١٢:٤٧:٠٦ م
تاريخ قصير:	١٤٢٥/١٢/٠٧
تاريخ طويل:	٠٧ ذو الحجة، ١٤٢٥

الدالة (GetChar)

إنها من الجديد المضاف فهي ترجع حرف من سلسلة تعين أنت رقمه فمثلا كلمة (فيجوال) هي سلسلة نصية أريد أن أجلب منها المحرف ٣ فترجع لي الحرف (ج)

```
MsgBox (Strings.GetChar ("٣" , "فيجوال" ) )
```

الفائدة منها أنت تحدهه فإذا كنت تريد على سبيل المثال أنت خزنت كلمة السر في سلسلة نصية وكان مكانها هو ٣ و٥ و٦ و٧ و٨ عن طريقها ترجع هذه المحارف

الدالة (Mid)

لهذه الدالة عمل مفيد فهي تقوم بعملية اقتطاع للسلسلة وتحويله لسلاسل أصغر منها أو حتى إلى سلاسل نصية ذات محرف واحد طبعا عندما وإليك مثال سيكفيك بإذن الله

```
Dim sName As String = "Visual Basic" ' السلسلة التي سنجري
' عملية اقتطاع عليها
Dim fCount As Short ' عداد الذي سنبدأ منه الاقتطاع
Dim fLen As Short = 1 ' عدد الحارف التي سنقتطعها
For fCount = 1 To 12
    ' تابع الكتابة من أجل رسم النص المقتطع
    Me.CreateGraphics.DrawString (Strings.Mid (sName, fCount,
    fLen), Me.Font.Brushes.Blue, fCount * 10)
Next
```

الدالة (StrReverse)

وهذه الدالة من الدوال الغريبة الاستخدام فهي تقوم بقلب السلسلة النصية بالعكس أي Learn تساوي عندها nrael ما عليك إلا أن تدخل المتغير النصي فيها لترى تحويلها وهذا الكود الذي يعمل عملها

```
Dim sELoad As String = "Do not Any things negative but
In job "
Dim fRe As String
fRe = ""
For a = 1 To Len (sELoad)
    c = Mid (sELoad, a, 1)
    b = Strings.Right (c, 1)
    fRe = b & fRe
Next
MsgBox (fRe) ' الكود الذي كتبناه
MsgBox (Strings.StrReverse (sELoad)) ' توفير الوقت من'
```

الدالة (StrComp)

لم يتم التغيير عليها في إصدار النت ولكنها من دوال المقارنة فهي تعود بثلاث قيم (٠ و-١) وهي تقوم بالمقارنة بين متغيران نصيان فعندما يكون الأول أكبر من الثاني فتعطي قيمة أكبر والصفر لتساويهم و-١ عندما يكون أصغر أما الفرق بين المقارنة بالنسبة لقيمة النصوص بالطريقتين نصيا أو ثنائي فسوف تراه عندما تكون قارنة بين سلسلتين متساويتين ولكن إحداهم أحرفها كبيرة والإخرى أحرفها صغيرة فعملية المقارنة من النوع

CompareMethod.Text

فترجع نتيجة عملية المقارنة بعد تجاهل الفرق بين الأحرف الصغيرة والكبيرة

CompareMethod.Binary

فهذا الخيار لن يتساوى بين قيم الأس كي لكل محرف أي الأحرف الكبيرة لن تستوي الصغيرة وفي حال إتباعه هذه الطريقة في كلمة المرور فسوف تجد مئات الكلمات أمامك

```
MsgBox (Strings.StrComp ("vb.net", "VB.NET",
CompareMethod.Text)) ' النتيجة ٠
MsgBox (Strings.StrComp ("vb.net", "VB.NET",
CompareMethod.Binary)) ' النتيجة ١ أي السلسلة الأولى أكبر
```

الدالة (StrConv)

ولعل دور عملية التحويل لها مكان ولكن من نواحي عدة فمثلا لنحول رموز اللغة الصينية التقليدية إلى رموزها الأصلية وجعل السلسلة النصية برمتها وخاصة للغة اللاتينية إ كبيرة أو صغيرة وأيضا نحبذ لو أن كل كلمة تبدأ بحرف كبير توفر لك هذه الدالة مجمل المتاعب عليك ما عليك إلا بإدخال رمز التحويل إليها وتراه وهذه أمثلة تجيبك عن بعض الإطروحات

```
Dim S As String = "all year and you are very good"
e.Graphics.DrawString (Strings.StrConv (S,
VbStrConv.LowerCase), Me.Font.Brushes.Blue, 1010)
' تصغير السلسلة النصية إلا أحرف صغيرة
e.Graphics.DrawString (Strings.StrConv (S,
VbStrConv.ProperCase), Me.Font.Brushes.Green, 1030)
' بداية كل كلمة حرف كبير
```

الدالة (StrDup)

هذه الدالة ليست جديدة فهي جاءت لتحل محل الدالة String\$ فكما تعلم فعندما قام صانعوا الإصدار الجديد طريق تسريع الترجمة \$ قاموا ولكن بقي شيء واحد أن معظم الدوال نفعت معها عملية فك علامة السحرية \$ ولكن الدالة String\$ أنت تعرف فعندما ألغوها تحولت لتعليمة مفادها أن المتغير هو قيمة نصية

Dim S As String

ومن هنا فكروا بإيجاد لها مكان فكان هو اختصار لمعناها الحقيقي Dup ولا أريد التفصيل فوظيفة هذه الدالة تكرر محرف ما عدد من المرات أنت تحده فكان لها أمجادها في الإصدار القديم وهذا مثال للإصدار القديم عليها عندما كانت String\$

```
Private Declare Function GetVolumeInformation Lib
"Kernel32" Alias "GetVolumeInformationA" ByVal
lpRootPathName As String ByVal lpVolumeNameBuffer As
String ByVal nVolumeNameSize As Long
lpVolumeSerialNumber As Long lpMaximumComponentLength As
Long lpFileSystemFlags As Long ByVal
lpFileSystemNameBuffer As String ByVal
nFileSystemNameSize As Long As Long
Private Sub Form_Load()

    Dim Serial As Long VName As String FSName As String
    VName = String$(255, Chr$(0))
    FSName = String$(255, Chr$(0))
    GetVolumeInformation "C:", VName, 255, Serial, 0, 0,
    FSName, 255
    VName = Left$(VName, InStr(1, VName, Chr$(0)) -
1)
    FSName = Left$(FSName, InStr(1, FSName, Chr$(0)) - 1)
    MsgBox "The Volume name of C\ is '" + VName + "',
the File system name of C\ is '" + FSName + "' and the
serial number of C\ is '" + Trim(Str$(Serial)) + "'",
vbInformation + vbOKOnly
End Sub
```

ولكن وضعت هذا المثال لأنك لن تستطيع إحصاره في لغتنا الجديدة لماذا لا أعرف هل لأنه يأخذ بطريقة مغايرة فقد تم إلغاء أو منع العشرات من دوال API أي من أن تعمل وذلك لتقليل مستوى لغتك من العبث بجهاز المستخدم وتم وضعها في أصناف أخرى شبيهة بالدوال

```
e.Graphics.DrawString(strings, StrDup(5, "m"), Me.Font,
Brushes.Blue, 10, 10) ' تصغير السلسلة النصية إلا أحرف صغيرة
```

الدالة (Split)

سأبدا بمدحها فهي دالة رائعة الاستخدام تفيد بحل معادلات الحساب وغير من مجموعات التجزئة والفائدة منها وفيرة إذا أحسنت الاستخدام... وظيفتها تعود بمصفوفة تحوي أجزاء سلسلة بعد إدخال عامل التباعد بين كل قيمة فأليك مثال نظري

```
Dim FName As String = "السلام عليكم ورحمت الله وبركاته"
```

إن هذه العبارة تحوي على سبعة أجزاء يفصل بينها فراغ وجزئين إذا كان الفاصل هو (س) وهكذا ولكن كيف سوف أقوم بتعريف المصفوفة إليك الطريقة نسند متغير ديناميكي أولاً دون

تحديد عدد أجزائه وطبعاً أرجو لن أشرح أنواع المصفوفات أو المتغيرات فعليك مراجعة كتب تخبرك عنها

```
DimFnameAs String="السلام عليكم ورحمة الله وبركاته"
DimSplit_Fname()As String= Strings.Split(Fname)
DimE_LenSplit_FnameAs Long
ForE_LenSplit_Fname=0 To
Information.UBound(Split_Fname)
MsgBox(Split_Fname(E_LenSplit_Fname))
Next
```

طبعاً تأكد أيضاً نستطيع جعل الفاصل نفس الحرف ومثلاً أكثر من مرة أي ** صحيح أنا قلت لك أن لها فوائد في حل معادلات الرياضيات المكتوبة فإلك طريقة بسيطة أولاً أنشئ صندوق إدخال وامنع المستخدم من إدخال غير الأعداد وفق قيم الأسكي ومن ثم هذه طريقة صغيرة وسأضع مثال عليها

3000+58252+44954+825448+96357+852417

هل عرفت ماذا سنفعل أولاً سوف نجزئ هذه السلسلة إلى مصفوفة باستخدام علامة "+" معادل التفريق ومن ثم نحولها إلى قيم عددية عن طريق الدالة Val وبعد ذلك نجعلها وتذكر المصفوفة ذات البعد الواحد

```
DimFnameAs String=
"3000+58252+44954+825448+96357+852417"
DimSplit_Fname()As String= Strings.Split(Fname, "+")
DimE_LenSplit_FnameAs Long
Dim النتيجة As Long
ForE_LenSplit_Fname=0 To
Information.UBound(Split_Fname)
النتيجة = النتيجة +
Val(Split_Fname(E_LenSplit_Fname))
Next
MsgBox(النتيجة)
```

الدالة (Join)

فلهذه الدالة عمل عكسي فهي تجمع المصفوفات وفق عملية إضافة أي معامل التفريق بينها، أنت سوف تضيفه ومعامل التفريق في هذا المثال هو *

```
DimFname()As String= {" استثمار"، "اللغة"، "بكل"، " "
"طاقتها"}
DimJoin_FnameAs String= Strings.Join(Fname, "*")
MsgBox(Join_Fname) ' استثمار*اللغة*بكل*طاقتها
```

الدالة (Replace)

أجزم أن رأيت هذه الدالة في كل من برامج معالجة النصوص فحين تريد البحث سيظهر لك خيار هل تريد الاستبدال بقيم أخرى وفي لغتنا تصور أنك كتبت أكواد في برنامجك وقد كنت أسميت أحد الأدوات اسم ما فأردت تغييره هل ستعيد كتابة كل الأكواد ولكنهم أضافوه أمرا في قائمة تحرير يسمح لك باستبدالها هل هذا صحيح ومن هنا تنطلق أهميتها

القيمة التي نبحث , السلسلة النصية التي سنجري عليها عملية الاستبدال (Strings.Replace)
رقم المحرف الذي سنبدأ منه , القيمة التي سنستبدلها بكل قيمة وجدناها , عنها بالسلسلة
(نوع عملية المقارنة, عدد مرات الاستبدال التي سنجرىها, البحث)

عمليتي المقارنة:

Binary ستكون ثنائيا أي المحرف (a) سيختلف عن المحرف (A) مما سيؤدي إلا عدم إدخاله
في عملية الاستبدال
Text ستكون العملية مقارنة نصيا أي A=a

```
MsgBox(Strings.Replace("All People love Vbnet", "e", "*",  
102, CompareMethodText))
```

الدالة (Filter)

بقيت محتارا في وصف هذه الدالة هل هي من دوال التنسيق أم من دوال البحث فتركت وصف اسمها حتى ترى نتائج عملها فهي تعود بقيم من مصفوفة نصية تكون من أجزائها فمثلا كلمة Sname هي كلمة وقيمة المراد إيجادها هي Sn من المأكد أن هذه القيمة تكون موجودة في أول السلسلة لذلك سيرجع الكلمة Sname كلها لأنه وجدها فيه وسيطها Include ذو القيمتين فاحذر منه لأنه إن كان يسوي False فإنه سيضيف كل نظير Match له نفس عدد المحارف إلا مصفوفة التحويل وإلا فن يشتمل التنظير على قيم ذات نفس عدد المحارف

والوسيط الأخير وهو نوع المقارنة فقد تم شرحه

أي هذا لتابع يبحث في مصفوفة ليعيد مصفوفة تحوي نظائر لنظير Match الذي أدخلناه وهذا مثال عليه

```
DimmyString$4) As String
    myString$0) = Learn
    myString$1) = "8
    myString$2) = End
    myString$3) = Rnd
    myString$4) = yt
DimrAs Short
DimsubStrings() As String= Filter(myStrings, "rn",
False, CompareMethod.Text)
'Include Value=false
'عيد المثال وغير قيمته إلى True
```

```
For r=0 To UBound(subStrings)  
    MsgBox(subStrings(r))  
Next
```

ستجد أن هذه الدالة ليست من دوال البحث المحرفي ولكنها تقوم بإيجاد قيم نظيرة لقيمة أردت البحث عنها ولم تجدها في سلسلتك وفي نهاية هذا الفصل إذا وجدت نفسك لم تفهم شيء فراجع الأمثلة فهي من أبسط الأمثلة استخداما وراجع المصفوفات لأنك إذا لم تفهما أو لم تستخدمها بعد فلن تفهم هذه الدوال التي تحتوي على وسطاء من المصفوفات

دوال الملفات والمجلدات

File System

سنأخذ في هذا القسم كيفية التعامل مع الملفات بالطريقة التقليدية إذ أن تم دعم
الفيجوال نت بطرق أخرى لن أخوض فيها
مثال File Stream
إذ سنأخذ نصيباً من وفير من التعامل مع الملفات

تم ترك معظم التعليمات في لغتنا إلى توابع حتى يتثنى للمستخدم كتابته وهي تقوم
محل
التوابع الأساسية ولكن السهولة في تقفي أثر الأخطاء أصبح أسهل والحال هنا نفسه
في توابع القراءة والكتابة من وإلى الملفات
فمثلاً هذه الطريقة المتبعة في فتح الملفات

Open "file.text" For binary as lock Write #1

أصبحت على شاكلة

FileSystem.FileOpen (1,"file.txt",OpenMode.Binary,OpenAccess .Write)

طبعاً لا أتكلم عن موجه الأوامر Cmd أو IO أقصد لن أشرح أوامر الموجه Console
والذي أصبحت أصناف ذات العدد ٣٧ ولكن واجهتني صعوبة في باديء الأمر وهي إن ضمن
الكتاب قصير على الأقل ولكن تراجع وتقلت سنبداً ولكن بطريقة غير عن الطرق الأخرى

أولاً: سأشرح البدايات

لماذا عمليات الكتابة والقراءة من الملفات عمليات مهمة ؟

كيف لك أن تخزن متغيرات عديدة ونصية كيف لك أن تخزن إدخالات السجلات التي أدخلها
المستخدم أن تحتاج إلى مكان تخزنه يمكن أن تستخدمه مرة أخرى غير الذاكرة التي ستفرغ
تلقائياً عند إغلاق برنامجك وليس هذا فقط إذا أردت معرفة خصائص الملفات الأخرى وقيمه
ومكان كسر كلمات السر البسيطة وهلم جرى
ومن هنا فقد تم أخيراً إحداث الصنف الذي يمكننا من كتابة أوامره ومن هنا نبدأ..

ثانياً : ماذا تغير

وقد تم تغيير عمليات استعمال الملفات إلى هذه الطريقة

FileSystem.FileOpen((آلية الوصول , طريقة الفتح, اسم الملف, رقم الملف)
-رقم الملف هو عدد من النوع long

-اسم الملف هو مساره كي تتمكن من الوصول إليه
-طريقة الفتح وهي 5 طرق

OpenMode.Binary ثنائي
OpenMode.Append عملية إضافة المتغيرات
OpenMode.Output عمليات إدخال المتغيرات
OpenMode.input قراءة المتغيرات
OpenMode.Random الوصول إلى الملفات بطريقة العشوائية عن طريق سجلات

أما آلية الوصول فهي عبارة عن طرق قفل إما تمنع من خلالها البرامج الأخرى من الكتابة أو القراءة أو كلاهما

OpenAccess.Read() للقراءة فقط'
OpenAccess.Write() للكتابة فقط'
OpenAccess.ReadWrite() للقراءة والكتابة'

أما طرق مشاركة الملف أثناء فتحه من برمجنا مع البرامج الأخرى

OpenShare.LockRead() قفل القراءة فقط'
OpenShare.LockWrite() قفل الكتابة فقط عن البرنامج الأخرى أثناء فتح الملف'
OpenShare.LockReadWrite() قفل القراءة والكتابة من البرامج الأخرى'
OpenShare.Shared() ملف مشترك

أم من أجل إغلاق الملفات فعلياً فتغلق بعبارة

FileClose((رقم الملف الذي فتحته)

واحذر أن تكتب الدالة Close لأنها ستغلق لك نافذة البرنامج

والآن سنبدأ طرق التعامل مع نوعين من فتح الملفات

Binary ,Random

ونبدأ مع النوع Binary

هذا النوع من الفتح وهو الفتح الثاني الذي سوف ندخل ضمنه قيم من البايت وحتى الكمية المعلومة وطولها فمثلاً أنت ستدخل القيم التالية بالترتيب الاسم وقد أدخلت خانة تساوي من عشرة محارف ومن ثم أدخلت العمر وهو من النوع Byte وأردت حفظهما فمثلاً قيمت الاسم لن تتجاوز العشرة بايتات وتحدها في صندوق النص ومادام العمر لن يتجاوز مقدار البايت الواحد ورقم البطاقة من النوع الطويل Long فسندخل البايتات بهذه الطريقة

```
Dim Xname as string, Age as Byte ,Sirral as Long
FileSystem.FileOpen(1,"c:\1.dat')
FilePut(1,Xname,1)
FilePut(1,Age,11)
File Put(1,Sirral,12)
```

وما تبقى فهي عمليات القراءة والكتابة
نستخدم

```
Print(مصفوفة المتغيرات , رقم الملف)
Write(مصفوفة المتغيرات , رقم الملفات)
```

نعم الفرق بأن الدالة Print ستضيف المتغيرات متلاحقة مما لا يثنى لنا معرفتها أما الدالة Write فهي ستضيف المتغيرات متلاحقة ولكن ضمن فهرسه كي يثنى لك إحضارها
مثلا:

```
Dims, dAs String
s = " mh"
d = "h"
FileSystemFileOpen(1,"c:\File.tex", OpenMode.Output)
FileSystemFileOpen(2,"c:\1File.tex,OpenModeOutput)
FileSystemPrint(1,s,d)
FileSystemWrite(2,s,d)
FileSystemFileClose(1,2)
```

أما هذه الصورة ستوضح الفرق بالإخراج

كما تعرف عندما نسحب نريد إدخال النص فلا فرق
ولكن المبرمجين قدموا الدوال الذي نريد والتي هي أيضا خاصيتين
بالعمليتين (Output,Append)
فالدالتين (PrintLine,WriteLine)

نستخدمها مع طريق الوصول (OutPut Append)

فهما تقومان بكتابة البيانات التي تريد تخزينها ضمنهما
أم الأمران

FileGet()

FilePut()



والآن سوف نتعلم حول التوابع الخاصة بالملفات مثل حجمه أو خصائصه أو أي مثل ذلك القبيل . أحب أن أنه أن الفيچوال بيسك ٦ لم يدعم المجلدات حق الدعم فقط أوجد بضعة توابع مثل إنشاء مجلد أو حذفه وإمكانية الحصول على خصائصه ولكنه لم ينجح في تغيير تلك الخصائص وكما نوهنا أنها كلها موجودة ضمن التابع الذي سوف توضحه الصورة
FileSystem

الدالة (FileLen)

وهي الدالة التي تسمح لك بمعرفة حجم ملف ما أنت تحدده عبر الوسيط الخاص بها مع العلم أنها تحضره بالبايت وفي حال أن الملف مفتوح ترجع لك قيمة صفر . وشكل هذا التابع هو

FileSystem.FileLen (Path As String)As Long

هو المسار الذي سوف تمرره إليها Path

وهذا المثال سوف يفي في الغرض

() حجم_الملفات Private Sub

& حجم الملف MsgBox

FileSystem.FileLen ("c:\windows\EXPLORER.EXE") & بايت

& " حجم الملف" MsgBox

FileSystem.FileLen ("c:\windows\EXPLORER.EXE") /1024&

" كيلو بايت"

' كي تعرف كم يساوي في الميغا قسم على ٢٤١٠٢٤ وهكذا على

نفس النحو

End Sub

' هذا المثال عبارة عن صندوقين متتالين سوف يحضران لك حجم

الملف المحدد في واحدتين مختلفان

الدالة (FileDateTime)

وهذا التابع من اسمه مبين ما هو . وهو من أجل معرفة تاريخ ووقت إنشاء ملف الذي تمرره لوسيط وطبعا تاريخ الإنشاء ميلادي فقط لا يمكن تغييره وشكله

FileSystem.FileLen (Path As String)

هو المسار الذي سوف تمرره إليها Path

وهذا المثال يوضح ذلك

& " تاريخ ووقت إنشاء الملف" MsgBox

FileSystem.FileDateTime ("c:\windows\EXPLORER.EXE"))

الدالة (File Copy)

وهذه الدالة جديدة و وظيفتها نسخ ملف ولصقه في مكان آخر مع الانتباه إلى أن الملف الذي يجب أن ننسخه يجب أن يكون موجود ضمن المسار المحدد والمكان الذي سوف ننسخ إليه الملف يجب أن يكون لا يحوي ملف ذو اسم مثل الملف الذي نسخناه وأن هذه الدالة لا تنسخ إلا ملف واحد فقط وهذا شكله

حيث أنه اسم الملف FileSystem.FileCopy (Source As String, Destination As String)
هذا الوسيط يعبر عن المسار الهدف الذي نريد أن ننسخ إليه الملف Source المراد نسخه Destination

وهذا المثال التالي يوضح عمل ذلك الملف

```
FileSystem.FileCopy("C:\Windows\EXPLORER.EXE",  
"d:\Mhdat")
```

الدالة (Kill)

وهذه الدالة من أيام البيسك الأولى وهي ومعناها أن تقتل أو احذف وهي كذلك لحذف الملف الممرر إليها ولكن يجب التأكد من المسار والشرط الآخر أن لا يكون الملف وضعه للقراءة فقط أو ملف نظام أو أي شئ من ذلك القبيل لأنها لا تستطيع حذفه وأحد مميزات الجيدة وهي أننا نستطيع حذف مجموعة ملفات في آن واحد وشكلها بسيط وهذا الكود يوضح عملها

```
On Error Resume Next  
FileSystem.Kill ("c:\Windows\Recent\*.*")  
'FileSystem.Kill ("c:\windows\Io.sys")
```

الدالة (Dir)

وهذه الدالة مفيدة من أجل البحث عن ملف معين أو ومجد نستطيع تعيين خصائص للبحث عنه وشكله التالي .

FileSystem.Dir (PathName, Attributes)

PathName هذا الوسيط لتحديد اسم الملف المراد البحث عنه

Attributes من أجل تحديد نوع البحث عنه وتقسيم

VbNormal....=0..... عادي

vbHidden...=2..... مخفي

vbSystem...=4..... ملف نظام

VBvolume...=8..... بتحديد هذه الخاصية يتم إهمال كافة القيمة الأخرى

VbDirectory..=16.... مجلد

vbArchive...=32.... أرشيف

vbReadOnly..=1..... للقراءة فقط

وتمكننا هذه الدالة من البحث على مجلد إذا حددنا الوسيط الثاني ١٦ وهكذا من أجل البحث عنه مع ملاحظة إذا أردنا البحث عن مجموعة ملفات حققت شرط الإدخال فتستطيع الدالة أن تعيدهم كلهم بجعلها وهذا مثال لذلك

```
Private SubSubSearch()
DimAdirAs String
DimAfiledirAs String
Adir = FileSystem.Dir("c:\windows", vbDirectory)
IfAdir = ""ThenMsgBox("لا يوجد مجلد نظام")
*****
' حيث يمثل رمز علامة التعجب أي حرف
Afiledir = FileSystem.Dr("c:\window$???.*")
Do WhileAfiledir <> ""
Print(Afiledir)
Afiledir = FileSystem.Dir' هذه ضرورية
Loop
End Sub
```

(GetAttr) الدالة

وهذه الدالة من أجل الحصول على خصائص الملف المحدد أو المجلد مثل للقراءة فقط أو مخفي أو ملف نظام أو ما شابه ذلك وتعيد ذلك رقمياً مما يتطلب ذلك أكواد من أجل معالجة وهذا مما يجعل الدالة الخاصة لحذف الملفات تساعد إذا ما حدث خطأ وشكلها بسيط فقط مرر مسار الملف الذي تريد أن تعرف خصائصه ويعيها أنها في حال الملف يملك خاصيتين تقوم بجمعهم معا ولتفادي ذلك نستخدم العبارة الشرطية والثابت وهذا مثال على ذلك

```
MsgBox(FileSystem.GetAttr("c:\BOOTSECT.DOS"))
```

(SetAttr) الدالة

وهذه الدالة عكس السابقة ووظيفتها تقوم بتعيين نوع الحماية للملف المحدد من جعله للقراءة فقط أو ملف نظام وهي مفيدة وأصبحت تدعم المجلدات على غرار السابقة مما يسمح بالقول أنه يوجد خطأ في الفيچوال بيسك القديم وهذا مثال يوضح ذلك

```
FileSystem.SetAttr "c\ mh.exe", FileAttribute.vbVolume -
FileAttribute.vbReadOnly
FileSystem.SetAttr "c\1", FileAttribute.System
حيث لا تعمل إلا إذا قمنا بطرح خاصية من خواصها أو أكثر
End Sub
```

(MkDir) الدالة

وهذه الدالة وظيفتها إنشاء مجلد في مسار نمرره لها ويجب الانتباه إلى أن يجب التأكد من المسار وصلاحيته عدم وجود مجلد يحوي نفس الاسم في المسار وإلى رسائل الأخطاء ولكي تجنب هذا الأمر نستخدم الدالة البحث وهذا مثال يوضح ذلك

```
Private SubSubSearch()
DimaFolderAs String
aFolder = FileSystem.Dir("c\MhAb", vbDirectory) ' هذه
السطر من أجل البحث عن المجلد في المسار اتالي
IfaFolder = ""Then
FileSystem.MkDir("c\MhAb")
Else
MsgBox("يوجد مجلد يحوي نفس الإسم في هذا المسار")
End If
End Sub
```

الدالة (Rmdir)

وهذه الدالة عكس الدالة السابقة بالتمام فهي تقوم بحذف المجلد الممرر إليه من فهرسه ولكنها تذكر يجب عدم الأخطاء في تحديد مسار المجلد ويجب أن يكون المجلد موجود وبس إلها حدة شغلة حلوة أنها تستطيع حذف المجلد مهما كان خصائصه (للقراءة فقط مخفي. أرشيف) مما يسهل علينا مهمة العناية ولكن لا يمكن حذف مجلد يعمل من داخله أي ملف أو برنامج وهذا التمرين سيفي في الغرض إنشاء الله

```
FileSystem.rmdir("c\Windows") حذف مجلد النظام
```

الدالة (ChDrive)

هذه الدالة تؤدي إلى تغيير السواعة الحالية يعني عند يعمل برنامجك طبعا يخرج لك المكان الذي يعمل منه برنامجك فأما هذه الدالة تقوم بتغيير السواعة لبرنامج ومع العلم هذه العملية وهمية إذ إن برنامج يعمل من السواعة الموجود فيه ولكن هذه إذا أنت أردت أن يعمل يشغل عدة برامج فإنك سوف تكتب المسار وكما تعلم قد يحدث عدة أخطاء ولكن هذه الدالة والتي بعدها تقوم بتغييرها مباشرة وما عليك إلا كتابة اسم البرنامج ليعمل أم المثال فسوف تجده في الدالة الثانية لتعلقهم بها وطبعا عندما تحدد اسم السواعة عليها أن تكون موجودة

الدالة (ChDir)

أما هذه الدالة فهي متممة للسابقة وتقوم بتحديد المسار الذي يلي اسم السواعة ويجب التنويه أن هذه الدالة لا تعمل لمفردها إذا ينبغي أولا تحديد اسم السواعة ومع العلم أنه هذه التعليمة تشبه تعليمة

Cd

الخاصة بالدوس ولكنها لا تختصر مثلها إذا يجب عليك كتابة المسار كاملا وهذا مثال يوضح عمل الدالتين السابقتين

```
Private SubChPro()
FileSystem.ChDrive("c\")
FileSystem.ChDir("c\windows")
```

```
Interaction.Shell("SCANDSKW.EXE", vbNormalFocus) ' تشغيل
تفحص الأقراص فقط كتبنا اسمه
'تصور أنه لديك مئات من البرامج فكم هذه الدالتين سوف
توفر عليك الكثير من كتابة المسار وأنتك تستطيع تغير المسار
الأساسي بهما
End Sub
```

الدالة (CurDir)

وهما دالتين تقريبا تعودان بنفس القيمة إذ تعودان بالمسار الحالي إذا لم يمرر لهما وسيط ويعودون باسم السواعة لهذا المسار الذي تم تخزينه ضمنهم ولكن لا يغيران الدليل أو السواعة الحالية. المثال التالي يفى بالغرض
واعلم أن هذه الدالة لا تحل محل الدالة App.path في الاصدار السادس فلذلك عملت مايكروسوفت على وضع مكتبة Microsoft.VisualBasic.Compatibility لان الدالة CurDir تتأثر في حال استخدمت أداة OpenFile وتأخذ القيمة لمسار هذا الصندوق ومن هنا سننشأ لديك مشكلة فإما اذا كنت لا تريد تغير المسار فاستخدم CurDir أما الدالة App.path فقاموا بايجاد الامر التالي بعد اضافة المكتبة المذكورة
VB.GetPath()

وهي عملية أكثر بكثير من الدالة CurDir

```
MsgBox(FileSystem.CurDir)
MsgBox(FileSystem.CurDir("f:\window\system")) 'output
F\
```

الدالة (FreeFile)

وهذه الدالة تقوم بتوليد أرقام للملفات المراد فتحها حتى لا يتعارض الأرقام مع بعضها ولكن ما دام الملف الذي تصنع لم يتجاوز مئة ميغا بايت فلن يحدث تعارض. ولكنهم خافو عليك ولمعرفة رقم الملف ما عليك إلا إسناد متغير عددي يسجل القيمة بداخله إذا آلية عملها كالتالي عندما تفتح ملف أول تقوم بإعطائه رقم ١ وهكذا وهذا مثال يوضح ذلك

```
DimNumberFileAs Integer
NumberFile = FileSystem.FreeFile()
FileSystem.FileOpen(NumberFile, "c:\1.txt",
OpenMode.Binary)
FilePut(NumberFile, "mbn#bkhjkb")
```

الدالة (Reset)

وهذه الدالة هي دالة شبيهة بتعليمية الإغلاق للملفات

Close

ومن دون أرقام إذا تقوم بإغلاق جميع الملفات التي تعمل من ذاك القرص التي قمت بفتح ملف من ملفاته وهي مفيدة وهذا مثال يوضح ذلك ولكن كلمة إغلاق غير صحيحة نسبيا

أغلقنا الملفات' FileSystem.Reset
 خطأ لأن الملف مقفول' (1, "mmmm") Print

الدالة (FileAttr)

وهذه الدالة جديدة على الفيجوال إذ تعود بعد إدخال رقم الملف إليها بنوع عملية الفتح التي قمت بها للملف ولكن بال أرقام وهذا الجدول يوضح ذلك

لذلك فما عليك إلا كتابة رقم الملف داخلها وهذا مثال يوضح ذلك

1	Input
2	OutPut
4	Random
8	Append
32	Binary

```
FileSystemFileOpen(1, "File.txt", OpenModeBinary
OpenAccess.Write)
MsgBox(FileSystemFileAttr(1)) 'output4
أغلقنا الملفات' FileSystem.Reset()
```

الدالة (Loc)

وتعود هذه الدالة بالقيمة معينة تمثل عدد البايتات أو الأحرف التي قمنا بكتابتها أو قراءتها من ملف قد قمنا بفتحه وتعود بالعدد ٠ إذا قمنا بفتح الملف وعدم الكتابة به والرقم ١ عندما نفتح الملف للقراءة ولا نقرأ منه شيء ملاحظة: تذكر أنها تعود برقم السجل المفتوح في حال كان نوع القراءة أو الكتابة عشوائي وهذا المثال الكافي إنشاء الله

```
FileSystemFileOpen(1, "File.txt", OpenModeBinary
OpenAccess.Write)
MsgBox(FileSystemLoc(1))
FilePut(1, "sdfsdf")
MsgBox(FileSystemLoc(1)) 'output6
أغلقنا الملفات' FileSystem.Reset()
```

الدالة (LOF)

هذه الدالة تشبه الدالة التي تحضر لنا حجم الملف ولكنها هذه الدالة تحضره بعد فتحه وفائدتها عندما نحن نريد القراءة من ملف معين ولنفرض إدخال كل محتواه ولكننا لا نستطيع معرفة طوله نقوم باستخدام هذه الدالة وإليك المثال التالي يوضح لك ذلك

```

FileSystem.FileOpen(1, "File.txt", OpenModeOutput,
OpenAccess.Write)
Print(1, "BGVFFD")
MsgBox(FileSystem.LO(1))
FileSystem.Reset() 'أغلقنا الملفات'
MsgBox(FileSystem.FileLen("c:\1.txt"))

```

الدالة (EOF)

وهذه الدالة تخبرنا إذا وصل مؤشر الكتابة أو القراءة إلى آخر الملف وترجع ذلك من نوع صح أو خطأ فإذا وصل التابع إلى آخر الملف القيمة ١ وإذا لم يصل القيمة ٠ أو خطأ وهذا المثال يوضح عملها

```

FileSystemFileOpen(1, "File.txt", OpenModeInput,
OpenAccess.Write)
Do Until(EOF(1) = True) 'الشرط هو أن يحضر السطور حتى يكون
وصل إلى آخر الملف
FileSystemLineInput(1)
Loop

```

الدالة (Seek)

وهذه الدالة ليس مثل الكلمة المحجوزة نفسها إن هذه الدالة فقط تعود بموقع مؤشر الكتابة أو القراءة ضمن الملف الحالي فأحد وظائفها أنك تريد ان ترتب قياس الفورم من طول وعرض وارتفاع و في ملف ثنائي ولم تعرف مواقع الكتابة بعد كل إدخال فتقوم هذه الدالة بإعادتها إليك وهذا يعني الموقع الذي سوف تبدأ منه القراءة أو الكتابة أكيد فهمت عليه لكنها لا تغيره وهذا مثال على ذلك

```

FilePut(1, "MsdnMh")
MsgBox FileSystem.Seek(1) 'موقع الكتابة التالي هو ٧

```

الدالة (Rename)

وأخير جاءت هذه الدالة لتلبي طلب المبرمجين بدلا من التعليمة الأساسية التي كان يجهلها الكثير ليس لصعوبتها ولكن لاختلافها

Name as

فهي تقوم بعملية إعادة تسمية (للملف أو المجلد) مما يسمح لنا ببناء مستعرض خاص يسمح لنا بنسخ وإعادة تسمية الأدوات والذي ستراه في أمثلة هذا الكتاب
فهي تقوم بتغيير اسمه فقط دون تغيير مساره أي لا تستطيع نقله لسواقة أخرى بالنسبة للمجلد أما الملف فهي تقوم بنقله للمسار الذي تراه

الدالتين (SPC, TAB)

وهذه الدالتين تستخدمان في من أجل عملية الإدخال من النوع Output للملفات النصية ولكن ولكن ستستغرب في كيفية عملها إذ أنك لا تدخل رقم الملف ولكن تصددهما بجوار الملف أي هما قيمة تستطيع استخدامها كفراغات وأعمدة إذ أنهم تضيفان فراغات وأعمدة أثناء الكتابة وهذه أمثلة على كل التابعين

```
FileOpen(1, "c:\1.txt", OpenModeOutput)
FileSystemPrint(1, "mhmsada$", FileSystemSP(10))
FileSystemPrint(1, "96598465")
FileClose()
```

ستلاحظ إضافة الفراغات في الملف النصي

```
FileOpen(1, "c:\1.txt", OpenModeOutput)
FileSystemPrint(1, "mhmsada$", FileSystemTAB)
FileSystemPrint(1, "96598465")
FileClose()
```

ستلاحظ إضافة مقدار عدة من الأعمدة بين البيانات المدخلة

(FileGetObject) الدالة

(FilePutObject) الدالة

(FileGet) الدالة

(FilePut) الدالة

هذه الدوال لنوعين من الكتابة العشوائية والثنائية وربما يقول الشخص ما هو الفرق بين كل زوجين على حدا والجواب وهو من كلامي لم يقر به أحد إذ إن الدالتين

Fileput,FileGet

تدعمان كل المتغيرات المطلوبة بينما الدالتين

FilePutObject,FileGetObject

تدعمان كل المتغيرات والمتمثلة بالنوع

Object

وأشبهه بأن يكون غرف من سيل والله أعلم ولكن إذا كانت نوع البيانات لاتهمك فاستعمل الدالتين المصحوبتان بالنوع Object

```
FileOpen(1, "c:\1.dll", OpenModeRandom)
FileSystemFilePutObject(1, MeWidth)
FileClose()
```

بينما لقراءة البيانات ستجد

Dim x As Object

```
FileOpen(1, "c:\1.dll", OpenModeBinary)
FileSystem.FileGetObject(1, x)
```

سيحجز أنه

بايت مقدارها قيمة

```
MeWidth = x
```

```
MsgBox (x)
FileClose ()
```

طبعا الحديث عن الفوائد التي لاتعد لعملية فتح الثاني والتي أعتبرها بداية الدخول نحو الكراكر طبعا فعندما يعرف أي مستخدم يعرف أن كل ملف كان عبارة عن بايتات مصفوفة بشكل متجاور) مع العلم أنها بكل مجازي هذه التعبيرات يفهم أن لكل ملف شكل فمثلا معظم (مترجمات لغات البرمجة) تترك القيم النصية على نفسها يعني

```
A="محمد"
```

وفي أي محرر ست عشري ستجدها كالعين المقلوبة تناديك إلي بوضع اسمك مكانها وذاك يعدو على الجميع

ولعل لم أقرب للشيء الذي أبحث عنه نعم لكل فمثلا لنذهب لملفات الصوت أو الفيديو فلو قلت في نفسك على سبيل المثال بالمؤكد ستقول الملف مؤلف من بايتات نعم وهذا صحيح وطبع لكل ملف منهجية تعتبر على الأقل مبدئية يعني تكون المعلومات عن الملف متوضعة عادة في أول الملف (كما في معجم الصحاح) تخبرك عن الملف إذا ما كان هل هو صحيح ملف صوتي أو أنك مخدوع بلاحقته فقط وعلى هذا ومثله وهذا مثال عملي لذلك

ضع هذا في منطقة التصريحات

```
Private Const SND_ASYNC=&H1
Private Const SND_FILENAME=&H20000
Private Declare Function PlaySound Lib "winmm.dll" Alias
"PlaySoundA" (ByVal pszName As String, ByVal hModule As
Long, ByVal dwFlags As Long) As Long
```

```
Function IfWave (ByVal FileNameWave As String)
    Dims As Byte
    Dim f As Byte
    Dim WordWave As String
    If FileExists(FileNameWave) = True Then
        Exit Function
    Else
        FileOpen(1, FileNameWave, OpenModeBinary)
        For s=9 To 12
            FileGet(1, f, s)
            WordWave = WordWave & Chr(f)
        Next
        FileClose(1)

        If WordWave = "WAVE" Then
            PlaySound(OpenFile.FileName, 0,
SND_FILENAME Or SND_ASYNC)
        Else
            MsgBox ("الويف نمط من ليس الملف هذا")
        End If
    End If
End Function
```

ومن ثم قم باستدعائه من أي مكان تريد

```
OpenFile.Filter= "Wave|*.wav"
```

OpenFile.ShowDialog)

IfWave=OpenFile.FileName هنا لتأكد التابع استدعاء يتم Wave ملف أنه من

ربما هذا الكود الأفضل في تشغيل ملفات الWave

ولكني لم أقتصر على هذه الحالة فلك أن تجد ضالتك عن طول الملف الصوتي أو أي ملف تريد فكما يجب أن تعرف أن لكل نوع وخاصة إذا كان مشترك خوارزميات أكواد طبعاً هنا الحديث ربما يفوق الحديث ولكن أرغب أن أعبر عن أهمية هذا القسم ولو كان بشكل مبدئي طبع الأمثلة كثيرة ومتعددة عن قسم الBinary طبعاً العمل في ملفات Binary أمر صعب بعض الشيء ولكن ممتع فمثلاً هذه الخوارزمية ممكن أن تحل أزمة

اسم المؤلف=15 Byte

تايخ الكتابة=8 Byte

مواليد المؤلف=8Byte

السيرة الذاتية للمؤلف=500 Byte

وعلى هذا المنوال ولكن فتقول لي طيب أحياناً يتطلب مني أن أجعل مجال النصوص محدود وربما سيكون المستخدم لن يدخل ١٥ حرف أقول استبدل كل حرف لم يكتبه المستخدم ب(*) على سبيل المثال:

وربما يكون مجال النصوص غير محدود وهنا أقول من الأفضل جعل القيمة النصية المفتوحة في آخر الملف وعليك بها في الزيادة وإن لا بد فعليك بنمط الفواصل أن تتبع مثال على ذلك

```
Dim xSp As String = "||||"
```

Dim As String هذه السلسلة غير محدودة الاسم

Dim As String هذه السلسلة الذاتية السيرة

محدودة

فإقرأ الثنائية القيم استخدام على مصر كنت إذا

التالي

```
FileOpen(1, "C:\MyFile.txt", OpenMode.Binary)
```

```
Text1.Text = الاسم
```

```
Text2.Text = الذاتية السيرة
```

```
FilePut(1, الاسم + xSp + الذاتية السيرة)
```

```
FileClose(1)
```

سيقول قائل وكيف أسترجمها أجيب راجع دالة Spilt بأن تجعل الوسيط الخاص لمحرك التجزئة هو "||||"

الدالة (FileWidth)

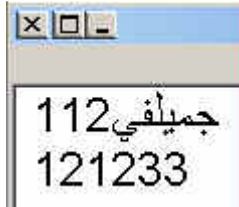
لم تكن سوى تعليمة أساسية أما الآن فهي عبارة عن تابع يستخدم مع عمليات فتح الملفات من النوع (Output) أي الإدخال النصي فهي تحدد عدد أحرف السطر الذي تريد إدخاله أي ما هي القيمة التي يجب أن يتألف منها سطر الإدخال ولكن عملية الإدخال هذه ستكون مختلفة أي إذا كنت تريد أن يجب عليك أن تكون قيمة الإدخال النصية يجب أن تكون مساوية

لقائمة العرض نفسها وفي حال كانت قيمتان الإدخال المتتاليتين أكبر من عرض الملف المدخل فإنه لن يقوم بعملية القراءة من سطر والكتابة في سطر بل سيضع كل مدخل في سطر

يعني ٥ محارف هذه يؤدي يجب أن تكون القيمة ٢ أو ١

```
FileOpen(1, "c:\yyyy.txt", OpenMode.Output) ' Open file
for output.
FileWidth(1, 10)
Print(1, "جميل")
Print(1, "في")
Print(1, "1")
Print(1, "12")
Print(1, "12")
Print(1, "1233")
FileClose(1)
```

هذه هي نتيجة التالي



وإذا كنت في ريب مما قلت فنفذ هذا الكود

الدالة (Input)

استخدام هذه الدالة لعله الأرجح في حال كنت قد قمت بكتابة مصفوفة من المتغيرات وتريد إرجاعها وخاصة إذا كنت استخدمت طريقة الكتابة

Write +++ Print

كما تعلم بعد فتح الملفات نكتب عادة

Write(1,S)

وكما تعلم ندخل مصفوفة من القيم

```
DimNameAs String= "أما"
DimIDAs Integer=541645
FileOpen(1, "c:\f", OpenMode.Output)
Write(1, NameID)
FileClose(1 )
```

ولكن لو سألنا أنفسنا كيف سنقوم باستدعاء ذلك كله فكما تعلم أن دالة

Input

لا تدخل لا ترجع سوى متغير واحد وليس مصفوفة كما في الإصدار السابق وهذا الكود المثالي لحل تلك الأزمة

```
DimNameAs String
```

```

Dim ID As Integer
FileOpen(1, "c:\f", OpenMode.Input)
Input(1, Name) ' سطر واحد قيمة لاستدعاء برجي
Input(1, ID) ' سطر واحد قيمة لاستدعاء برجي
MsgBox(Name & Space(5) & ID)
FileClose(1)

```

ونفس الأمر مع الدالة إدخال

Line Input

الدالة (LineInput)

استخدام هذه الدالة مقتصر على نوع واحد من طرق الفتح وهو

Input

أي فتح الملفات النصية وتقوم بإحضار سطر واحد فقط ومع ذلك فإنها مفيدة على كل زعم وهذه الطريقة لسحب كل أسطر الملف المفتوح

```

FileOpen(1, "C:\MyFile.txt", OpenMode.Input)
Dim x As String
Do Until EOF(1) = True
    x = LineInput(1)
    MsgBox(x)
Loop
FileClose(1)

```

الدالة (InputString)

هذه الدالة من الجديد المستحدث في الإصدار الجديد ولكنها من الفائدة تجل محل عملية قراءة الدفتر أي بدلا من استخدام الكود في الدالة السابقة

ومادام النص الذي تحمله يعبر عن متغير واحد فيمكنك حينها من إدخال النصوص الكتابية كما بالطريقة التالية (أقصد أن ما تم كتابته في الملف هو عبارة عن نص فقط أو رقم بشاكلة نص) والآن ما علينا إلا معرفة حجم الملف الذي نريد فتحه

```

FileOpen(1, "C:\MyFile.txt", OpenMode.Input)
Dim x As String

x = InputString(1, FileLen("C:\MyFile.txt"))
' ReadOnly بقسمة فتحته الذي الملف تقفل لم حال في عليك
' الملف فتح قبل الملف بحجم وتعبيته Long النوع من متغير بتصميم
MsgBox(x)
FileClose(1)

```

(Lock) الدالة
(UnLock) الدالة

هاتان الدالتان مرتبطتان مئة بالمئة وعليك استخدامهما بالشكل

Lock()

...

Unlock

وتقومان بعملية قفل الملف وفتحه ولكن ليس كما تظن مثل دالة

Close

ولكن وظيفتهما وهي تأمين بعض العمليات التي تحتاج للوصول لنفس الملف المفتوح وهما تقومان بهذه المهمة حسب ما أخبرت به مايكروسوفت

```
FileOpen(1, "c:\ptt.txt", OpenMode.Binary)
    Lock(1)
    FilePut(1, "MoreLearnForPeople")
    Unlock(1)
    FileClose(1)
```


دوال المعلومات

Information

الدالة (VarType)

لعل القديم قد احتوى على مثيلها فهي تقوم بإرجاع نوع المتغير فمثلا لو أدخلت له اسم النافذة
me رجعة Object لان النافذة هي من النوع Object
وهي ترجع رقم يمثل نوع المتغير فمثلا الرقم 8 يعبر عن نوع المتغير string و لكن أنت غير
مطالب بحفظه فكما تعلم الصنف المنسق له جاهز لراحتك فإنه سيحواله إلا اسم نصي مباشر
MsgBox (Information.VarTypeMe) 'outout9

```
MsgBox (Information.VarTypeMe.ToString) 'output Object
```

الدالة (VarName)

هذه الدالة نفس السابقة إلا أنها ترجع نوع المتغير نصي أي ترجع اسمه لكن اعلم أن كلا
الدالتين السابقتين منفصلتين بالأداء

```
MsgBox (Information.VarNameMe) 'outout Object
```

```
MsgBox (Information.VarTypeMe.ToString) 'output Object
```

الدالة (VbTypeName)

عمل هذه الدالة غريب بعض الشيء فهو يرجع متغير نصي. ويقوم بعملية مقارنة مع أي نوع
للبيانات مثلا المتغير

```
Dim X As String "Long"
```

```
MsgBox (Information.VbTypeName(X)) 'outoutlong
```

ما بين علامات التنصيص يحول إلى نوع متغير ولعل الفائدة منه هي إيجاد كم مرة تم إيجاد هذه
الكلمات لأنواع المتغيرات فعلا سبيل المثال في صناعة برنامج محرر للفيجوال بيسك سكر بيت
فإنك أفضل من أن تدخل مئات من جمل الاختبار لأي كلمة تستطيع تغير لونها مادامت لا ترجع
قيمة "

الدالة (SystemTypeName)

هذه الدالة جديدة على لغتنا ولعلها لا تفيد إلا في ناحية واحدة وهي معرفة نوع التعريف
بالمغيرات بالنسبة للنظام حيث تقدم لنا اسم نوع البيانات فتعيده لك بشكل النظام التابع للغة
يعني القيمة Long مساوية لنوع Int64 من حيث النوع الذي تم دعمه أي معظم النتائج ستكون

هي نفسها فلا تستغرب ما عليك إلا إضافة الاسم الصحيح وستعطيك النتيجة

```
DimShAs String= "Short"
DimDaAs String= "Date"
DimNoAs String= "eeee"

DimNameTypeAs String
NameType = SystemTypeName (Sh) ' يعود
"SystemIn#0.
Msgbox(NameType)
NameType = SystemTypeName (Da) ' يعود
"System.DateTime".
MsgBox (NameType)
NameType = SystemTypeName (No) ' لاشيء
MsgBox (NameType)
```

الدالة (IsArray)

يعني ستتحقق من المتغير من أي نوع كان فإذا كان هو عبارة عن مصفوفة سترجع الأمر True وإلا False
وبجدر الاهتمام أن نذكر فائدة غير تلك الفائدة فمثلا أنت تريد تشكيل مصفوفة لاسم شخص وعمره وتوالده وتريد أنه فعل تلك المصفوفة في وقت طلب الأمر تفعيل

Dim F() AS object

هنا سترجع قيمة IsArray القيمة False ولكن عندما ستفعل قيم لها سترجع لك قيم صحيحة
ReDim F(3)

أي ستكون قيمة التحقق True

الدالة (IsDate)

هذا معامل اختبار إذا كانت القيمة الممررة إليه تاريخ فأعطيه علامة صح بمعنى العبارة ولكن كم تنسيق يسمح هذا المعامل بقبوله فكثير أي مهما كانت أشكال الفوارق فإنه يقبل العديد من التنسيقات مما يجعله غير آمن في بعض الحالات فسوف أذكر لك بعض الحالات التي يقبله

```
DimFAs Object
F = April1 /2004
F= 1-1-2004
F=1 \ 1 \ 2005
F= April1 2003
F= "1 nov2004"
F=#1/12003#
```

لذلك صحيح أنك لن تستطيع كتابة التاريخ في أحد الطرق المذكورة ولكنها تعد جميعا تحت نوع المتغير Date

الدالة (IsNoting)

هذه الدالة أيضا من دوال الاختبار (صح // خطأ) أما هذه الدالة تختبر أي متغير من النوع Object إذا كان شيء موجود أو أنه غير موجود مثلا على

```
Dim s As Object
MsgBox (Information.IsNothing(s)) 'output True
```

هنا نوع المتغير خاص بالنوع الكائنات

الدالة (Is Numeric)

هذه الدالة تختبر نوع المتغير إذا كان عدد أو لا أي قيمة عددية حتى ولو كانت ضمن علامات تنصيص أو من أي نوع كان

```
Dim f As Object
Dim s As String
f= 5458
s=548518
MsgBox (Information.IsNumeric(f)) 'output False
```

وإياك أن تفرن هذه الدالة بالدالة Val لأن الدالة Val تم شرحها أما الدالة السابقة فهي ستخبرك إذا كان القيمة الممررة إليها عدد فترجع قيمة قبول ومن هنا تستطيع رفض أي قيمة غير ذلك ولعلها تفيدك في أماكن اختبار القيم

الدالتين (LBound, UBound)

إن لهذه الدالتين شغل قديم فهما تقدمان عملا واحد لقيمتين متعاكستين. وظيفتهما مقتصرة على المصفوفات فالدالة LBound اختصار LowBound فتعني أقل محدد للمصفوفة وأما Ubound فهي اختصار لUpBound وهي أعلى قيمة محدد داخل المصفوفة الفائدة كثيرة في إجراء العمليات على المصفوفات من جمع إلى قلب إلى إسناد المنفصلان أي آخر محدد أضفته للمصفوفة وليس له علاقة بقيمه ف وتمعن هذان المثالان

```
Dim F(3) As Long
F(0) =10
F(1) =7000
F(2) =-500
F(3) =6000
MsgBox (Information.UBound F) ToString 'output 3
```

عملية جمع المصفوفتان

```
Dim F(3) As Long
F(0) =10
F(1) =7000
F(2) =-500
F(3) =6000
Dim Co As Short Equals As Long
For Co =Information.LBound F To
Information.UBound(F)
    Equals = Equals + F(Co)
Next
MsgBox (Equals)
```

الدالة (Err)

هذه الدالة أقرب لأن تكون صنف وأظن أنها عبارة Structure فكما تعلم فإنه قد يحوي على توابع ضمنه على أية حال إن لهذا الكائن فوائد فهو يقدم لنا الآتي في حال وقعنا في الخطأ ولن أقوم بشرح عمليات تتبع الأخطاء لأن هذا الكتاب لا يتطرق لذلك ولكن إليك التفاصيل

On Error Resume Next

تذكر هذه العبارة فهي سنقوم بعملية اختبار عليها في أمثلتنا القادمة ومعناها البرمجي (في حال وقوع الخطأ اذهب إلا السطر الذي يليه ومنها فستقوم بملء التوجيه ERR بالمعلومات من أجل خواصه ولنبدأ

Clear-١

فهذه المنهج فسوف يفرغ الدالة Err من المعلومات لأنه سوف يقوم بحذف الأخطاء التي خزنة فيه أصلاً ولن يرجع لك أي معلومة وتستعمل بعد إيجاد أو إعطاء التفاصيل عن الخطأ وتخزينه حتى يتم زيادة سرعة البرنامج بعض الشيء

Err.clear

Description-٢

كما نعلم أن يجب على كل شخص فعل فعل غير صحيحة أن يوضحها فأيضاً لكائن الخطأ وصف يجب توضيحه ومنها جاء عمل هذه الدالة فلكل خطأ وصف ويمكنك أيضاً إسناد وصف خاص معين بالخطأ

Number-٣

لعله كان علينا أن نبدأ بها قبل الوصف لكن لاضير فكما تعلم أنه يجب على الحوادث أو الأخطاء أن تكون مرتبة فلذلك يجب أن تكون مفهومة وفق دليل يخبرنا برقم الخطأ ومن أهم تلك الأشياء بناء ما يسمى بأصناف أو سجل تسجل به أخطاء التي تحدث وترفقها مع برنامج حتى يسهل عليك إرسال رقع برمجية تسد الخلل فلذلك أنصحك بأن تجعل جميع التوابع والمكونات في مكتبات خارجية حتى يسهل عليك تجنب الأخطاء وإصلاحها فيما بعد

On Error GoTo 100

DimgAs Byte

g = InputBox(أدخل عمرك)

Exit Sub

:١٠٠

'ملاحظة يمكنك أيضاً التحكم برقم الخطأ إذا كنت تتوقع حدوثه

IfErrNumber=6 Then

Err.Description = إن هذا القيمة غير ممكنة

MsgBox(Err.Description)

End If

Err.Clear()

الدالة (QBColor)

أصغي جيداً...تغير مفهوم الألوان من أنواع الدعم الثلاثي إلى مفهوم جديد إلى ٦ قيم (أحمر- أخضر- أزرق-- و التدرج --والإشباع--والإضاءة)فعندما تريد تنفيذ الكود التالي

MeBackColor=QBColor(10)

ستشاهد تسطير تحت تلك الدالة لتعبر عن خطأ لماذا أو لا وحلها ثانياً؟

تم جمع الألوان بصنف خاص أو كائن هو Color وبعد ذلك جعلوا إسناد الخصائص تابعة لتلك

Proprety BacColor as Color

End Property

أي علينا تحويل تلك القيمة المنطوية تحت إطار Integer إلى ذلك الصف وفق تابع خاص لن تستطيع إضافته إلا من الشركة الأم على سبيل تحويل الإطار العام ولكن عملية التحويل متشكلة بعدة أنواع وهذه هي إحدى الطرق ولكن عمليا هذه الدالة تحوي على ١٥ لون أساسي تراها في معظم البرامج البسيطة وأيضا كانت تدعمها لغة البيسكا القديمة أي ٠ يعطي أسود وهكذا أي (GDI)

```
Dim f As ColorTranslator
```

```
Me.BackColor = f.FromOle(QBColor(1))
```

```
Me.BackColor = f.FromWidth(FromOle(QBColor(2))) ألوان
```

نظام

لاحظ أنه يمكن أيضا بعدة طرق أي كل طريقة ترجع قيمة بالتحويل لاصناف الكتابة

الدالة (RGB)

أرجع لشرح للدالة السابقة وافهم مقدماتها ومن ثم أرجع لها الآن فكم تعلم أن الألوان الأساسية التي تظهر لك في شاشة الحاسب والتي يدعمها كرت الشاشة ٣ ألوان (أحمر-أخضر-أزرق) وكانت هذه القيم تتراوح بين ٠ إلى ٢٥٥ لكل قيمة (R,G,B) حيث من أجل الحصول على ألوان فمثلا القيم

0,0,0 ستعطي اللون الأسود أما 255,255,255 فتعطي اللون أبيض ولم تتغير قيمتها كما تغيرت الآن فقد صارت مسألة الألوان مدعومة أكثر وعمليا لتحول هذه قيمة هذه الدالة من النظام

Long

لتوافق نظام أصناف الألوان راقب هذا المثال الذي سيغير لك لون الفورم في كل مرة تحمل البرنامج لون جديد باستخدام الدالة السابقة

بدء عملية التحويل 'Dim f As ColorTranslator

```
VBMath.Randomize()
```

هذه العبارة هي التي لاتسمح للفورم ' أن يظهر بنفس اللون في كل مرة

```
Me.BackColor = f.FromWidth(Information.RGB(255 * Rnd(),
```

```
Rnd() * 255, Rnd() * 255, Rnd() * 255))
```


الأمر لا تحتاج إلى تعقيد فالدالة ربما ستكون مألوفة وتستطيع التحكم بقيمتها لأنها توجه قيمتها لعملية إدخالها

الدالة (MsgBox)

فأما هذه الدالة فهي أخت الدالة InputBox في لغة البيسك وقد قمنا باستخدامها بشكل كثير فهي أكثر الدوال المستخدمة في شرح الامثلة فأما شرحها عبارة عن مثال بسيط

هنا شرح سوف يظهر على سطح الصندوق (MsgBox هنا " MsgBoxStyle.Critical + MsgBoxStyle.YesNoCancel , عنوان الصندوق
أما ما تبقى باللون الأزرق فهو عملية التحكم بمظهر الصندوق وكم من الأزرار نريد ان يظهر

أما هذه الطريقة فلتحكم بأزرار الصندوق بطريقة جديدة

```
Dim f As MsgBoxResult
f = MsgBox("إظهار من أجل إظهار", MsgBoxStyleCritical + MsgBoxStyle.YesNoCancel,
"شاشة إظهار")
Select Case f
Case MsgBoxResult.Cancel
MsgBox ""
Case MsgBoxResult.Yes
MsgBox ("نعم للإسلام")
Case MsgBoxResult.No
MsgBox ("لا للكفر")
End Select
```

الدالة (IIF)

لهذه الدالة المستحدثة من الإصدار السادس عمل جملة If في حال التمييز بين قيمتين وفق شرط ثابت

```
IF Nu > 500 Then
True"الإخراج النصي أو قيمة
Else
False"الإخراج النصي أو قيمة
End if
```

يعني نضع الشرط أولاً ووفق هذا الشرط ستظهر لنا إحدى القيم إما True—False أو قيم معينة لأن هذه الدالة من النوع Object أي تقبل كل القيم بداخلها القيمة التي ستحملها الدالة , القيمة التي ستحملها الدالة في حال كان الشرط صحيح , الشرط (IIF) (في حال كان الشرط خاطئاً)

ويمكنك التعامل مع الشرط كما تريد وبراحتك

```
Dim Fals As String
Dim f As Long
f = InputBox("أدخل العدد")
Fals = IIf(f > 20 And f < 50, "غير مقبول", "مقبول للوظيفة مرحليا", "مقبول للوظيفة مرحليا")
MsgBox(Fals)
تجنب هذه الفكرة لأنها ستظهر لك القيمتان مع بعضهما
Fals = IIf(f > 300, MsgBox("مقبول"), MsgBox("صغير"))' صندوقان لكل
قيمة صندوق
```

تجاهل القيمة التي سيخرجها لك محرر الدالة لوجود الخطأ فيها للسبب

الدالة (Chosse)

نعم لهذه الدالة المضافة في الإصدار السادس وهي من أجل إختيار قيمة عبر Index من مصفوفة من أي نوع . فعلى سبيل المثال تصور وهي أقرب لعبارة SELECT وفق شخصية واحدة من وجهة نظري. اعلم أنه سيبدأ العد من الرقم واحد

```
Dim f As String
f = Interaction.Choose(, " كبير", " متوسط", " صغير")
MsgBox(f)
```

وأما بالشكل التالي أدخل أي رقم صالح في تلك المصفوفة ذات البعد الواحد

```
Dim f(2) As String
Dim h As String
f(1) = "ملا"
f(0) = "عبد الناصر"
f(2) = "خطيب"
h=Interaction.Choose(1, f)
MsgBox(h)
```

المهم أن تعلم انها دالة إختيار قيمة من مصفوفة عامة القيم وبالتالي أقرب للتعليمة Select لقيمة Index عددية من النوع المضاعف

الدالة (SaveSetting)

هذه الدالة من دوال التفاعل مع الرجستري في نظام الويندوز. حيث تقوم بحفظ قيمة في مفتاح وفق مكان محدد على المسار

```
HKEY_CURRENT_USER\Software\VB and VBA Program Settings
وسيتم إنشاء فهارس التالية
```

وسأقوم بوضع الشرح المندرج (مشروح كل جزء فيها)

إسم المجلد الذي سيلبي المسار الذي أخبرتك عنه" ، " SaveSetting (" المجلد الثانوي الذي سيلبيه" ، "إسم المفتاح" ، قيمة التي سيتم (إضافتها للمفتاح

فكما تعلم الدالة مقيدة شيء ما من حيث المكان التي ستظل فيه ونوع البيانات المدخلة ولكن السبب؟؟

هو أنه ستلغي الغرضية الدعائية من توابع Api التي سمح المبرمجين والمبرمجيات من التحكم بالرجستري والفائدة منها هي حفظ الإعدادات في الرجستري وهذا مثال

```
Dim FName As String
FName = InputBox("أدخل اسم")
Interaction.SaveSetting("MCCN", "mhmam", "Style", FName)
```

الدالة (GetSetting)

الآن جاء دور إحضار المعلومات من الرجستري وأقصد من المسار التالي

KEY_CURRENT_USER\Software\VB and VBA Program Settings

بحيث المعلومات التي أدخلناه من الدالة السابقة نحضرها أولاً للدالة نفس شكل الدالة في كل من الوسطاء ماعدا الأخير حيث استبدل بقيمة Default حيث هي القيمة الافتراضية بحال لم تجد الدالة المفتاح وهي مفيدة وتجنب الأخطاء من أجل التحكم بالخصائص بشكل مقبول

```
MsgBox (Interaction.GetSetting("MCCN", "mhmam", "Style",
"محمد")
```

الدالة (DeleteSetting)

فأما هذه الدالة لعلها تستخدم في حال برامج التنصيب وإزالتها حتى تقوم بجعل رجستر المستخدم نظيف ولا يحوي أي قيمة ما أما من أجل الوسطاء فهي نفس التي سبقتها أي (الرئيسي- الثانوي -المفتاح المراد حذفه) وعندما تكتب المفتاح الرئيسي فقط فإنه سيحذف مجلد الإعدادات بأكمله وإذا كتبت الرئيسي والثانوي سيقوم بإبقاء الثانوي وهكذا

```
Interaction.DeleteSetting "MCCN", "mhmam", "STyle") ' حذف
المفتاح
Interaction.DeleteSetting "MCCN", "mhmam") ' حذف الفهرس
الثانوي
Interaction.DeleteSetting "MCCN") ' حذف الفهرس الرئيس
بأكمله
```

ملاحظة مسار البيانات التي هو خاص بكل مستخدم مع العلم في حال أردت استخدام الدوال السابقة

ولم تكن مثبت الفهرس

HKEY_CURRENT_USER\Software\VB and VBA Program Settings

فالطامة الكبرى سوف تحدث ورسائل الأخطاء ستلاحقك هذا ما حدث في الإصدار القديم أما في الإصدار الجديد ربما أبدلت (ميكروسوفت) بدالة ذكية

الدالة (GetAllSettings)

هل تريد على الحصول على كل شيء موجود في مسار المسجل الخاص وعلى القيم التي تحتويها المفاتيح . تكرر عندما قلت لك أن المصفوفات الثنائية لن نلزمنا إلا بدالة واحدة وهي هذه الدالة فالصف الأول منها يعيد اسم المفتاح بمسار مسجلك المحدد والصف الثاني يرجع ما يحتويه مفتاحك من قيم

```
DimTAs Long WAs Long
DimX (,)As String= GetAllSettings ("Tec", "Mccn")
ForT=0 ToUBound(X)
ForW=0 ToUBand(X)
Msgbox (X(T, W)) ' ترجع المفتاح و قيمته
Next
Next
```

الدالة (Shell)

هذه الدالة قديمة أيضا وهي من أجل تشغيل البرامج التنفيذية أو الطلب من برامج تنفيذية تشغيل برامج من اللاحقات الأخرى أما ما أضافه مبرمجوها هو ما يسمى مصطلح الانتظار مع تحديد مدة الانتظار أي مهلة ريثما يقوم البرنامج بتنفيذ أوامره ومن ثم التحكم فيه إن شئت عن طريق الدالة SendKeys أو ما هناك

وتكمن جمالية خاصية الانتظار فمثلا تريد أن تنصب برنامج معين ولكن لانريد أن يظهر النافذة الحامل للرقم التسجيل إلا لينتهي برنامج التنصيب فتقدم لك هذه الدالة الخاصية بأريحية (لن ينتقل التركيز لنافذتك حتى ينتهي البرنامج من عمله

```
Interaction.Shell "C:\Setup.exe",
AppWinStyle.NormalFocus, True
```

لن ينتقل التركيز للبرنامج حتى ينتهي من عمل البرنامج الآخر ولعل هذه الفائدة تفيد في كثير من الأحيان أم إذا كنت تريد تحديد وقت محدد حتى تستطيع العمل في برنامجك ما عليك إلا تحديد وقت وهو بالملي ثانية

```
Interaction.Shell "C:\Setup.exe",
AppWinStyle.NormalFocus, True, 1000
```

أما أوضاع الاستدعاء فهي كالتالي

```
AppWinStyle.NormalFocus تشغيله بوضعه العادي
AppWinStyle.Hide وضع مخفي
AppWinStyle.MaximizedFocus وضع كبير
AppWinStyle.MinimizedFocus وضع مصغر
MinimizedNoFocus بدون تركيز على البرنامج
```

ولكن إذا كانت حالة البرنامج في حالة الانتظار فلن تفيد كل الحالات ما عدا المخفي منها

الدالة (AppActivate)

هذه الدالة أمرها غريب فلقد كانت في الإصدار القديم تفيد بالغرض المطلوب لها بدون الحاجة لعنون البرنامج الذي سيظهر في قائمة المهام

```
Dim f As Integer
f = Shell "c:\AthanBasic.exe")
AppActivate f) لن تعمل
```

هذه الطريقة غير فعالة أبدا كما في الإصدار القديم في هذا المثال
أما طريقة عملها فهي أن لكل برنامج يعمل في الذاكرة ما يسمى عنوان غير اسم الملف



هذا اسم الملف الموجود على الجهاز



المطلوب وخاصة للملفات التنفيذية حصرا
أما للواحق الأخرى فلا داعي لذلك ما عليك إلا كتابة اسم الملف نفسه الموجود على الهارد

سوف تعمل لكل اللواحق التي ' (Athan Steup.txt)" AppActivate
تعمل في الذاكرة

سيقول قائل لماذا الإطالة
فأقول : كثير من البرامج تفيد في كسر أقراص الاوتران الحاوية على أفخاخ برمجية أو غير ذلك

الدالة (Command)

إن الكثير من شاهدها ومنهم القليل الذي عرفها ووظيفتها تكمن أهميتها بدعمها لمفاتيك ولواحقك التي تحملها

فمثلا أنت صنعت برنامجا لقراءة الملفات النصية وحفظها وليس من المعقول أن من يستخدم برنامج يجب عليه تشغيل البرنامج أولا ومن ثم يحدد الملف المطلوب حفظه أليس ذلك
فما ضير أنه عندما يريد نقر النقرة المزدوجة على أي ملف نصي إذا كنت مسجله بالرجستري أو فتح بواسطة ومن ثم تحدد مسار برنامجك مما يؤدي إلى جعل برنامج هو برنامج التشغيل لمات نصية ولكن بعضهم قال فتح الملف يكون ولكن لا تتحمل أية بيانات إلى صندوق النص وأجيبهم أنهم لم يستعملوا هذه الدالة

التنفيذ:

عندما تقوم بفتح ملف أي ملف بواسطة برنامجك تحمل المسار المطلوب إليها كاملا وكما قلت لك عندما تستخدم



إن الصورة أوضح
بعد ذلك سوف تشحن الدالة (Command) بمسار الملف النصي
على سبيل المثال

"C:\ملف نصي.txt"

والآن جرب الكود التالي في الحدث التحميل للفورم Load

```
'عليك إضافة صندوق نص اسمه TextBox
Dim f As Long
If Interaction.Command <> "" Then
f = FileLen Interaction.Command)
```

```

MsgBox Interaction.Command)
FileOpen 1,InteractionCommand
OpenMode.Input)
TextBoX.Text=FileSystem.InputString 1,f)
FileClose 1)
End If

```

وجرب الطريقة بالصورة السابقة بعد عمل Build للبرنامج

سوف يصفك البرنامج ويحذرك أحلامك



لاحظ أنه تم تكرير عملية التنصيب مرتين

""C\txt""

مع أنه في صندوق النص أعطاك السبب لن أدخل وأقول لماذا ولكن حلها بعملية الكود مع التعديل

```

TextBoX عليك إضافة صندوق نص اسمه '
DimfAs Long
DimXAs String
IfInteraction.Command <> ""Then

Msgbox Interaction.Command)
X = Strings.Right(Interaction.Command,
Len(Interaction.Command)- 1)
X = Strings.Left(X, Len(X)- 1)
f = FileLen(X)
FileOpen1,X,OpenModeInput
TextBoX.Text=FileSystemInputString1,f)
FileClos1)

```

وتنجز المهمة وابتسم دائما فأنت مبرمج فيجوال بيسك (أقوال الأخ تركي العسيري)

الدالة (Environ)

كثيرون من صغروا من دواننا ومن أعماها بالفعل أن من الحزينين على ما آلت إليه أحوالنا فهل تعلم يا أخي أن والحمد لله تستطيع أن تحصل على اسم النظام ومنصته والسواعة الرئيسية عناوين هامة في نظامك دون أن تكتب كلمة من API وبالإصدارين نعم في السادس كانت

موجودة وابقى عليها في السابع نعم هذه الدالة تمثل معظم موارد نظام التي يقوم بها من العمليات الموجودة فيه فهي تعيد من الرقم ١ إلى ٣٢ من الخدمات بما فيها نوع المعالج وعائلته فكل رقم يقابله معلومة من معلومات النظام فالرقم ١٥ يعطي منصة النظام الحالي

```
Msgbox Environ(15) 'out put OS= .....
```

أو

```
Msgbox Environ ("OS")
```

أي تستطيع كتابة الوسيط بطريقتين فاختر الأسهل ولكي تحصل على جميع المعلومات من الناحية العملية

```
Dim FCount As Byte
```

```
For Fcount = 1 To 32
```

```
Listbox1.items.add Environ(Fcount)
```

```
Next
```

الدالة (CallByName)

هذه الدالة قلما من يستخدمها فيه من الاسم معناها الاستدعاء بواسطة الاسم ولكن هذه الدالة قلما تفهم

فهي من الشكل العام

```
CallByName (ObjectRef, ProcName, UseCallType,  
ParamArrayArgs)
```

ObjectRef

هذا القسم فهو لتحديد اسم الأداة البرمجي مثل Label , TextBox1

ProcName

إسم الخاصية المراد استدعاء ها

UseCallType

نوع الاستدعاء حسب الخاصية (إذا كانت للقراءة أو تقبل التعديل أو أن تكون منهج)

CallType.Method خاصة بالمنهج

CallType.Set خاصة بالخصائص التي تقبل التعديل

CallType.Let نفس الأمر لمعظم الخصائص

CallType.Get خاصة بالخصائص التي هي للقراءة

وهذا مثال:

```
CallByName Me "Text", CallType.Set, "الاسم الجديد"
```

(اللفورم)

```
CallByName Me "Hide", CallType.Method)
```

الدالة (Partition)

هذه الدالة مهمة للغاية فهي تعيد الفئات من الأرقام حسب إدخالتك فمثلا أنت تريد أن ترجع لك تدخل لها رقم ٢٠٠٦ وهو المحصور بين ٢٠٠٠ و ٢٠٥٠ حيث الفارق الذي يفصل بن هذان العددان هو ١٠

ولتبسيط عليك مثلاً أدخل عمرك وقل لهذه الدالة أن ترجع لك الفئة هل أنت مواليد الثمانيات أو التسعينيات أو الستينات أو الألفين (ممكن تكون صغير قليلاً)
فتقول هل أنت مواليدك ١٩٨٦ وهي محصورة ١٩٧٠ وبين ٢٠٠٠ والقفز سيكون ١٠ سيقول لك أن هذا التاريخ محصور بين ١٩٨٠ و ١٩٨٩ وهو المطلوب أي بالثمانيات وهذا المثال:

```
Dim year As Long = 1986
Dim decade As String
decade = Partition(year, 1950, 2050, 10)
MsgBox("السنة " & CStr(year) & " عقد في إنه " & decade & ".",
MsgBoxStyle.MsgBoxRight + MsgBoxStyle.MsgBoxRtlReading)
```

(GetObject) الدالة

(CreateObject) الدالة

هاتان الدالتان أقوم بشرحهما في هذه الطبعة حتى لا أتطرق وأدخل بحرب سؤال وجواب على المكونات إلى أنني أريد أن أنوه أن المكتبات التي تكون CLSID والتي لا يستطيع مضيف الإضافات إضافات توابعه فلن تستطيع الاستفادة منها

```
Const ForReading = 1, ForAppending = 8
Dim objFSO, objTxt
objFSO = CreateObject("Scripting.FileSystemObject")

'إنشاء الملف النصي'
objTxt = objFSO.CreateTextFile("C:\Test.txt", True)
objTxt.WriteLine "السطر الأول"
'سطين فارغين'
objTxt.WriteLine (2)
objTxt.Write "السطر الرابع"
objTxt.Close

'لإضافة فتح الملف'
objTxt = objFSO.OpenTextFile("C:\Test.txt", ForAppending)
objTxt.Write vbCrLf & "السطر الخامس"
objTxt.Close

'فتح الملف للقراءة'
objTxt = objFSO.OpenTextFile("C:\Test.txt", ForReading)
'قراءة كافة البيانات'
msgbox objTxt.ReadAll
objTxt.Close
```

والحمد لله رب العالمين والصلاة على رسول الله وعلى آله وصحبه
أجمعين

٢٠٠٦/١/١

mccn@gawab.com

