

C# .NET 2005

إعداد وتنسيق: y85.fox

تأليف: حمادة النهضة

الفهرس

الموضوع	الصفحة
مقدمة.....	5-
من نحن؟.....	6-
النتيجة.....	7-
الدرس الأول.....	9-
مفهوم لغات البرمجة.....	9-
مفهوم الكائنات.....	11-
تطبيق الدرس الأول.....	12-
واجب الدارس الأول.....	18-
الدرس الثاني.....	19-
المتغيرات.....	19-
تطبيق الدرس الثاني.....	23-

مقدمة:

السلام عليكم ورحمة الله و بركاته...

النتيجة: // صنع برنامج مشغل صوتيات شبيه بالريال بلاير و الجت أوديو .

الهدف: // إخواننا الكرام في منتدى الفريق العربي للبرمجة .

الطريقة: // دروس برمجة مشروحة و معروضة على شكل صور موضح عليها كل شيء .

ما يميزنا عن الجميع: //

- 1- أننا نجيب على كل الأسئلة بإذن الله .
- 2- أننا سنواصل معكم حتى النهاية ولو بقي معنا شخص واحد فقط.

أمثلة لنتائج من طبقوا تلك الدروس

من نحن ؟

حماة النهضة مجموعة من الإخوة متحابين في الله وضعوا نهضة الأمة نصب أعينهم و يسرون لها بشتى و مختلف الطرق سواءً أكانت إيمانية , أو تكنولوجية , أدبية , ثقافية , برمجية بكل الطرق و في هذا الدرس نسعى جاهدين لإفادتكم فأنتم إخواننا.

يا إخوة هذه الدروس مبتدئة لكن بنفس الوقت هيا رائعة و سنتابع بعدها دروساً أكبر و أضخم إن شاء الله .

من رأى فينا خطأً فلينصحننا فنحن بشر

لسنا الوحيدون و لكننا نسعى للأفضل بإذن الله لخير الأمة و نهضتها

و إنه لأملٌ أملٌ جهدٌ و عمل

صناع الثقة و التغيير

حماة النهضة



1- النتيجة:

باختصار الموضوع عبارة عن مجموعة دروس مبسطة في لغة C# 2005 للمبتدئين

بهدف إنشاء هذا البرنامج لتشغيل الصوتيات والمرئيات :



الدروس ستكون مقسمة إلى قسمين , قسم الدرس وقسم التطبيق .وفي كل درس نبدأ بشرح الفكرة أو المفهوم أو القاعدة وفي قسم التطبيق نطبق اللي تعلمناه من الدرس على مشروع البرنامج.ونستمر بهذه الطريقة كل درس إلى أن ننجز المشروع.

لكن قبل البداية يجب تجهيز برنامجين للعمل على اللغة والبرنامج الأول من ميكروسوفت وهو برنامج ملفات لغة البرمجة

.NET Framework

[للتحميل من هنا](#)

حجم الملف يقارب 24 ميجا بايت

عند فتح هذا اللينك ستجد زر مكتوب عليه Download



البرنامج الثاني هو بيئة التطوير , أي المكان الذي سنكتب فيه تعليمات البرنامج ونجربه .
يوجد هناك عدة بيئات تطوير مجانية متوفرة منها بيئة ميكروسوفت بإصدارها المضغوط Express

لتحميل C# 2005 Express

حجم الملف يقارب 450 ميغا بايت

إذا لم تستطع تحميل هذه البيئة فهناك بيئة أخرى مجانية

Sharp Developer

حجم الملف يقارب 4 ميغا بايت

لتنظيم العمل على الدروس ينصح عمل مجلد جديد على القرص المحلي D وتسميته CS

يجب أولاً تنصيب البرنامج الأول dotnetfx.exe

وعند الانتهاء منها يتم تنصيب البيئة SharpDevelop2_2.0.0.1591_Setup.exe أو Express #C

وبهذا تكون مستعداً لتلقي الدرس الأول



الدرس الأول

مفهوم لغات البرمجة

منذ بداية ظهور الكهرباء وتطورها , ظهرت فكرة معالجة التيار الكهربائي والاستفادة منه .فمثلاً المصباح يأخذ الكهرباء

ويعالجها "يحولها" إلى ضوء .والراديو يأخذ الكهرباء ويحولها إلى صوت .. وهكذا

وبالتالي فإن أي عمل مع الكهرباء يحتاج إلى ثلاثة أشياء مهمة

1- مدخلات "تيار كهربائي".

2-معالجة "تحويل التيار إلى ضوء , صوت....".

3-مخرجات "ضوء , صوت ..".

بالإضافة إلى جهاز التحويل "المصباح , الراديو"

الحاسوب لا يخرج عن هذه القاعدة فهو يحتاج إلى مدخلات ليعالجها ويحولها إلى مخرجات.

مثلاً عندما تضغط أي زر في لوحة المفاتيح , فإن تيار كهربائي يتولد وينطلق باتجاه المعالج.

المعالج بدوره يحول هذا التيار إلى تيار مختلف ينطلق نحو الشاشة ويظهر الحرف .قد يتساءل البعض كيف يفرق المعالج بين زر وآخر في لوحة المفاتيح , فجميع الأزرار تطلق تيار كهربائي. وهذا هو أساس العمل في الحاسوب , حيث أن معالج الكمبيوتر لا يستقبل تيار مستمر. كما في المصباح , أو في الراديو , أو في أي جهاز كهربائي وإنما معالج الكمبيوتر يستقبل تيار متغير .

للتوضيح نفرض أن العدد 1 يمثل مرور تيار والعدد 0 يمثل عدم مرور التيار , عندما يكون المصباح مطفأ يكون التيار الذي

يستقبله هكذا

0000000000000000

أما عندما نضيء المصباح فإن التيار يكون هكذا

1111111111111111

وهكذا مع الراديو و المكواة و...و....

لكن في الحاسوب يختلف شكل التيار لأنه يكون متغير فيكون شيئاً هكذا

110001010100011110101000111010101001

وهذه هي الطريقة التي يفرق المعالج حرف عن حرف آخر فالتيار الذي يمثل حرف الواو مثلاً يكون هكذا

11001010 وحرف الطاء يكون هكذا 10111010 وحرف النون يكون هكذا 01010011

وبالتالي فإن كلمة وطن ستكون هكذا

010100111011101011001010

ولا ننسى أن معنى 0 عدم مرور تيار و معنى 1 مرور تيار وبافتراض أن سرعة قراءة المعالج لرقم هي ثانية واحد
 فأن التيار الكهربائي الذي يمثل حرف الواو 11001010 سيكون هكذا:
 في البداية ينقطع التيار لفترة ثانية ثم يمر التيار لمدة ثانية, ثم ينفع لمدة ثانية ثم يمر لمدة ثانية ثم ينقطع لمدة ثانيتين ثم
 يمر لمدة ثانيتين , أي أن المعالج سيسغرق ثمان ثواني لقراءة حرف الواو .
 في المثال السابق افترضنا أن سرعة المعالج هي ثانية واحدة لكنها في الحقيقة أسرع بكثير , ومن هنا اعتقد أننا فهمنا ما
 معنى سرعة المعالج "هي سرعة قراءته للرقم الواحد", والرقم الواحد سواء كان 1 أو 0 يسمى "بت" Bit
 فعندما تكون سرعة المعالج 3200 يعني أنه يقرأ 3200 ميغا بايت في الثانية الواحدة , والميغا يحتوي على 1024 كيلو
 بايت والكيلو بايت يحتوي على 1024 بايت , والبايت يحتوي على 8 بت , يعني أن هذا المعالج يقرأ 3277832 بت في
 الثانية الواحدة . يعني أنو يقرأ حرف الواو في مدة 0.00000244 من الثانية .
 والآن بعد أن عرفنا كيف يقرأ الكمبيوتر الكلمات بقي أن نعرف كيف ينفذ الكمبيوتر الأوامر المطلوبة منه. كان الحاسوب في
 بدايته يستقبل الأوامر كتيار كهربائي مثل ما رأينا سابقاً , لكن هذه الطريقة غير عملية , فإذا أردنا أن نأمر الكمبيوتر بأن
 يطبع كلمة وطن على الشاشة فسيكون الأمر شبيهاً بهذا :

0101001110101010101010100111011101011001010

فهذه هل اللغة التي يفهمها الحاسوب , لكنها لغة صعبة الفهم على الإنسان , لذلك ظهر ما يسمى بلغات البرمجة
 " وهي لغات قريبة من لغة الإنسان ويفهمها الحاسوب"
 حيث أن لغات البرمجة سهلت التعامل مع الحاسوب وأصبح من الممكن إصدار الأوامر للحاسوب هكذا:

كود

Print "وطن"

وحيث أن الحاسوب لا يفهم هذه اللغة يوجد ما يسمى بالترجم , وهو برنامج ملحق بلغة البرمجة وظيفته قراءة الأوامر
 المدخلة وتحويلها إلى تيار يفهمه المعالج , لكن عملية الترجمة تتم بشكل غير ظاهر , فما على المبرمج إلا أن يكتب الأوامر
 ويشغل البرنامج .. والمترجم يقرأ Print "وطن" ويحولها إلى :

0101001110101010101010100111011101011001010

ويرسلها إلى المعالج..



مفهوم الكائنات

في لغات البرمجة كل شيء يعتبر كائن له صفات وله أفعال كأى كائن في الوجود , فمثلاً السيارة كائن له صفات "مثل اللون والشكل وعدد الإطارات...." وله أفعال مثل "السير , التوقف, ..."

لا حظوا أن الأفعال قد تكون من الكائن نفسه "السير" و قد تكون من كائن آخر , في الحاسوب أيضاً الأمر لا يختلف فكل شيء يعتبر كائن له صفات وأفعال , فمثلاً هذا المنتدى كائن له صفات وأفعال , و زر "اقتباس" تحت هذه المشاركة هو كائن له صفات مثل اللون , و الكلام المكتوب عليه و الحجم... وله أفعال مثل أنه يعمل رد جديد مع اقتباس الرد الحالي... و صندوق النص في آخر هذه الصفحة عند الرد السريع , يعتبر كائن له صفات كاللون وله أفعال كإرسال الرد إلى قاعدة بيانات المنتدى..

زر ابدأ أو Start في الوندوز هو كائن وشريط المهام Task Bar هو كائن يحتوي على كائن آخر "زر ابدأ" إذاً يمكن لكائن أن يحتوي على عدة كائنات فالسيارة كائن يحتوي على كائن آخر وهي الإطارات مثلاً .

تطبيق الدرس الأول:

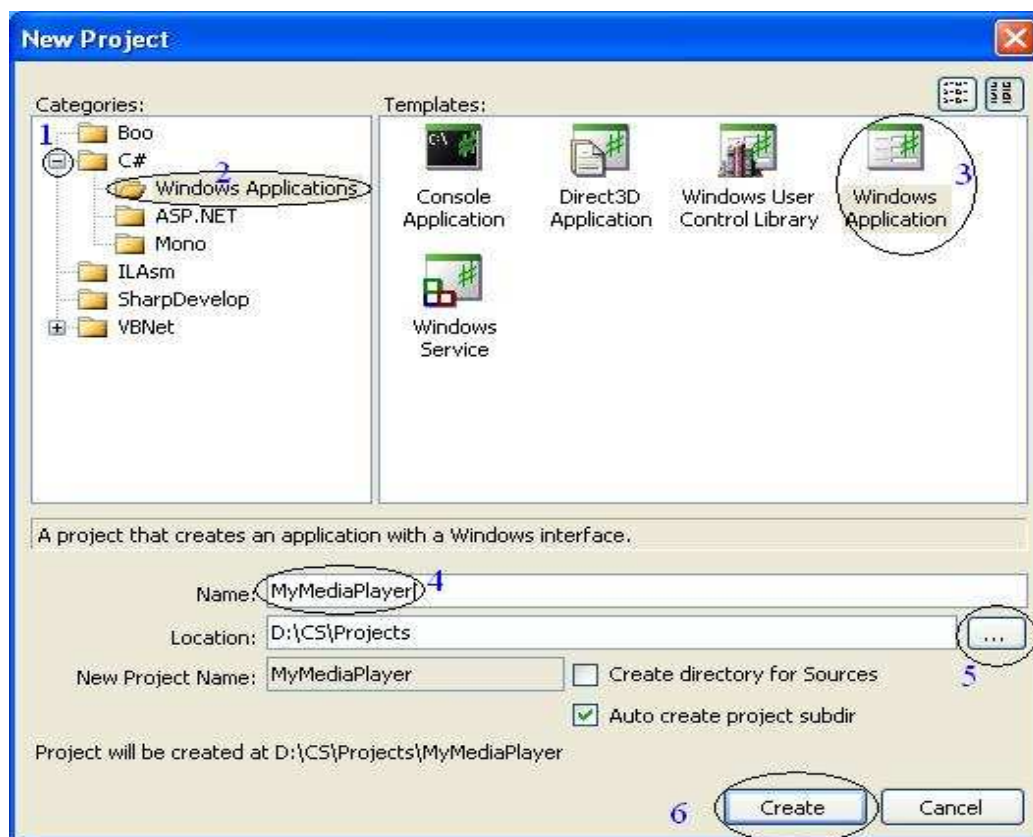
بعد تنصيب البرامج افتح مجلد CS الذي أنشأناه في القرص D وداخل المجلد CS قم بإنشاء مجلد جديد اسمه Projects من سطح المكتب لتجد إيقونة برنامج بيئة التطوير وهذا شكلها:



بعد أن تفتح البيئة سيظهر لك زر في الوسط هكذا :



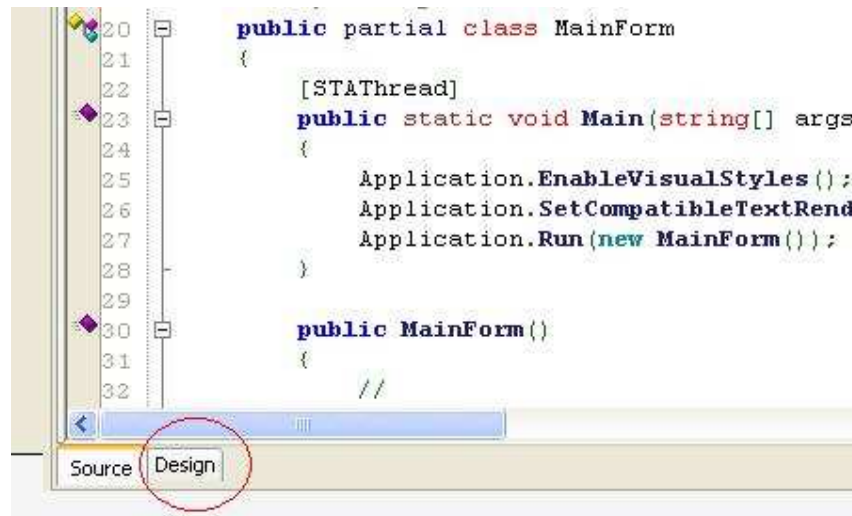
أنقر على زر New Solution المؤشر عليه بالأحمر ستظهر لك هذه النافذة:



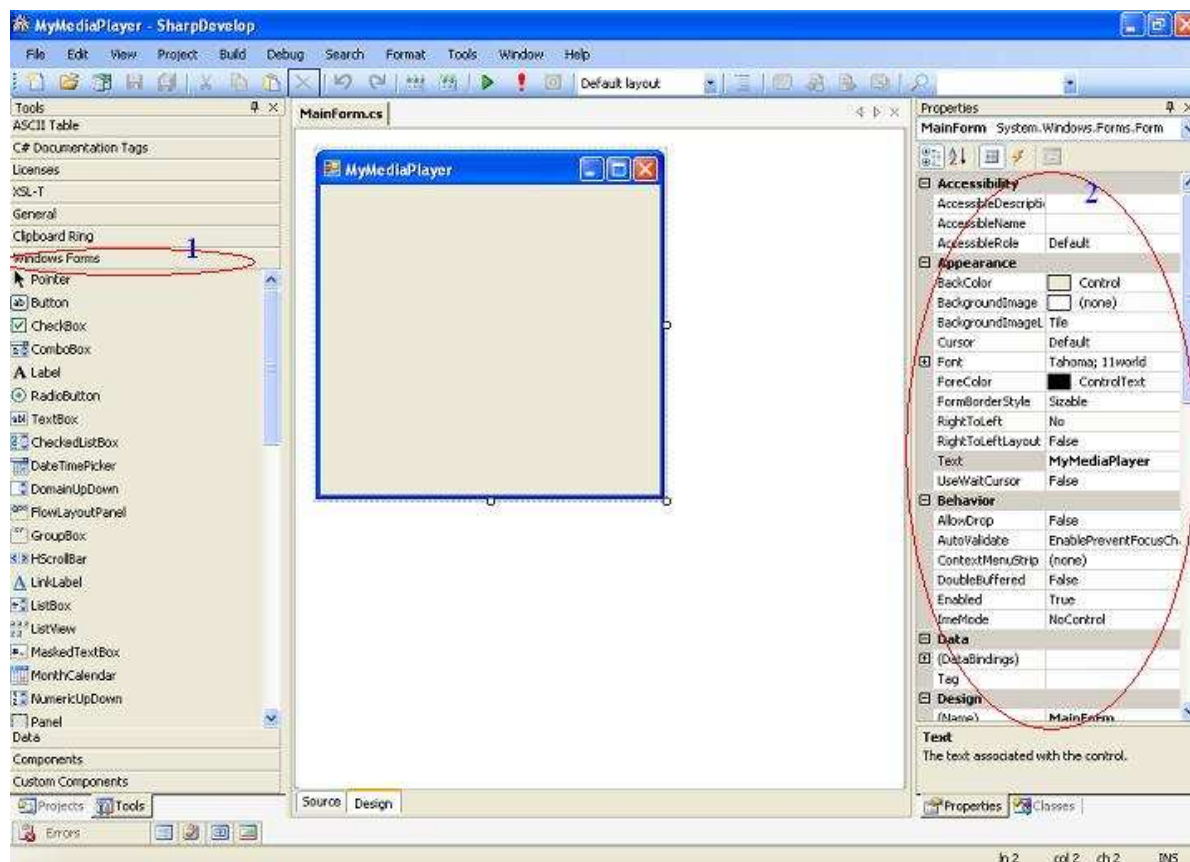
أنقر لى إشارة + الموضحة في رقم 1 ثم انقر على Windows Applications كما في رقم 2
ثم أنقر على Windows Application كما في رقم 3 , عند صندوق اسم المشروع أكتب اسم المشروع الذي ترغب
به وليكن:

MyMediaPlayer كما في رقم 4 ثم انقر على الزر في رقم 5 وإختر مجلد Projects

الذر أنشأته سابقاً داخل مجلد CS في القرص D ثم أنقر على زر Create في رقم 6
تظهر لك نوافذ المشروع وفيها شفرة المشروع الافتراضية , تحت نافذة الشفرة انقر على زر Design كما في الصورة



ستظهر أمامك هذه النوافذ:



في اليسار انقر على Windows Forms كما في رقم 1 ستظهر لك جميع الكائنات التي نحتاجها في البرامج , وفي الوسط ستظهر لك نافذة فارغة نسميها Form وهذه النافذة هي النافذة الرئيسية للمشروع , ولا ننسى إن هذه النافذة تعتبر كأن لها صفات مثل أسمها "MyMediaPlayer" ولها أفعال مثل ما سنلاحظ لاحقاً , على اليسار يوجد جدول وهذا هو جدول صفات الكائنات ومن خلاله يمكن تغيير أي صفة من صفات الكائن , ما رأيك لو قمنا بتغيير لون النافذة إلى الأزرق.. نذهب إلى جدول الخصائص ونبحث عن خاصية BackColor , ننقر على الصندوق أمامها ستظهر قائمة ألوان , نختار منها أي لون وليكن الأزرق

نلاحظ أن لون النافذة تغير , يعني أننا غيرنا صفة من صفات هذا الكائن , الآن نرجع الكائن للونه الطبيعي

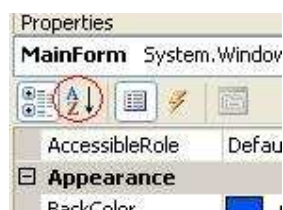
يوجد صفة أيضاً مهمة أسمها Text أبحث عنها في جدول الصفات عند رقم 2

ستجد مكتوب أمامها MyMediaPlayer

أمسحها وأكتب بدلها "مشغل الصوت والفيديو".

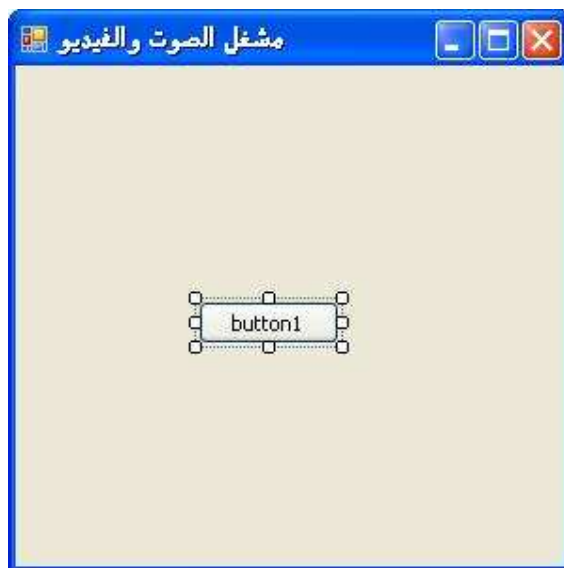
ملاحظة: لسهولة الوصول إلى الصفات يمكن ترتيب الصفات بالترتيب الأبجدي بالنقر على زر الترتيب في أعلى جدول

الصفات :



من نافذة الأدوات على اليسار انقر على صورة الزر المكتوب أمامه **Button** , واستمر في الضغط واسحب الماوس فوق النافذة الفارغة .

سيكون شكل النافذة هكذا:



يمكن تغيير حجم أو موقع الزر , الآن تأكد من اختيار الزر و اذهب إلى نافذة الصفات , أبحث عن صفة اسمها **Text** وأكتب بدل **button1** أكتب "ترحيب".

لاحظ أنا غيرنا صفة من صفات الزر , ولكن كيف نغير من أفعال الزر لأنه كانن ولا بد له من أفعال , والزر يمكن أن يفعل أي شيء لكننا نريده أن يظهر رسالة ترحيب ..

للذهاب إلى أفعال الزر أنقر نقرًا مزدوجًا **Double Click** على الزر ,

ستظهر لك نافذة الشفرة :

.

.

```

35 InitializeComponent
36
37 //
38 // TODO: Add const
39 //
40 }
41
42 void Button1Click(object sender, EventArgs e)
43 {
44 }
45
46 }
47
48

```

وينتقل مؤشر الماوس إلى مكان بين حاصرتين مكتوب فوقه **void Button1Click** ومعنى هذا أن المعالج عند ضغط الزر سينفذ الأوامر المكتوبة بين الحاصرتين , لا تهتم للشفرة المكتوبة فوق ولا تعدل فيها أبدا فهي مهمة لتشغيل البرنامج الآن بين الحاصرتين يجب أن نكتب الأمر الذي يظهر الرسالة وهذا الأمر هو:

كود

```
MessageBox.Show("مرحباً بكم إلى سي شارب");
```

لنشرح الأمر , أولاً **MessageBox** هو كائن في لغة البرمجة يمثل إظهار رسالة و **Show** هو كائن آخر داخل كائن **MessageBox** ويقوم بطباعة الرسالة على الشاشة وكلمة "**مرحباً بكم إلى سي شارب**" تعتبر مدخل للكائن وهو يعالج الأمر ويخرج الرسالة على الشاشة .

لاحظ أن الأمر **MessageBox.Show** يعني أظهر رسالة , ولكن المعالج لن ينفذه لأنه لا يعرف ماذا يكتب في الرسالة لذلك يجب فتح قوس وكتابة الرسالة بين علامة اقتباس , والقوسين تعني أن بينهما مدخلات , أما علامة الاقتباس فهي مهمة لأنه بدونها سيعتبر المترجم أن مرحباً بكم إلى سي شارب هي جزء من الأمر .
ولا يفهم هذا الأمر لذلك لن يعمل البرنامج ويجب أن تكون الرسالة بين علامتي اقتباس حتى يتجاوزها المترجم, في نهاية السطر يجب كتابة الفاصلة المنقوطة حتى نخبّر المترجم أن هذه هي نهاية السطر وينتقل إلى السطر التالي .

ملاحظة: يجب أن تكون حذراً جداً عند كتابة الشفرة فزيادة نقطة أو حرف تجعل المترجم يرتبك ولا يشغل البرنامج وأيضاً يجب مراعاة الأحرف الكبيرة والصغيرة

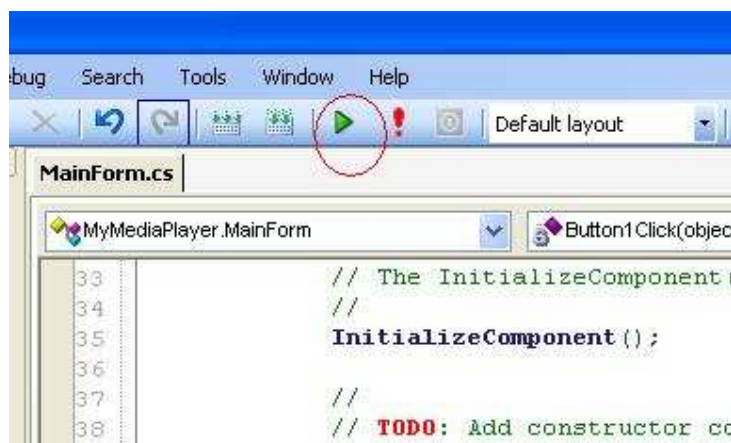
فالمترجم لن يفهم هذا الأمر messagebox بينما يفهم هذا الأمر MessageBox

.

.

.

بقي لنا أن نشغل البرنامج ونرى كيف سيعمل , لتشغيل البرنامج انقر على زر التشغيل أعلى نافذة الشفرة:



عند تشغيل البرنامج ستظهر لك النافذة الأساسية وعند النقر على زر الترحيب ستظهر رسالة الترحيب:



مبروك لقد نجحت في تنفيذ أول برنامج بلغة C#



واجب الدرس الأول

أنشئ مشروع جديد و أنشئ فيه زر لرسالة ترحيب , غير لون النافذة الأساسية , و غير حجمها , و غير لون الزر و غير حجمه , من الممكن أيضاً إنشاء زر آخر يظهر رسالة أخرى وأرسل التطبيق .

لإرسال التطبيق:

اذهب إلى مجلد CS في القرص D ثم إلى مجلد Projects ثم إلى مجلد MyMediaPlayer

ثم إلى مجلد bin ثم إلى مجلد Debug ستجد ملف تنفيذي بإسم MyMediaPlayer

هذا هو برنامجك الأول .. أضغطه ببرنامج الونزيب وأرسله لي

ملاحظة: عند عمل الواجب أو عند تجربة كائنات أخرى , أنشئ مشروع جديد للعبث به , لكن لا تغير أي شيء في المشروع الذي نطبق عليه

نهاية الدرس الأول.



الدرس الثاني

المتغيرات

في الدرس الأول تعلمنا أن كل شيء في الحاسوب يعتبر كائن وفي الحقيقة هناك نوعين من الكائنات :

كائنات حقيقية: وهي التي لها صفات وأفعال ويمكن أن يكون لها تمثيل مرئي.

متغيرات : وهي متغيرات لا تحتوي على صفات أو أفعال وإنما تحتوي على قيمة محددة ولا تكون مرئية , أي أنه من الممكن أن يحتوي الكائن "**المتغير**" X على قيمة مخزنة في ذاكرة الحاسوب , ومن الممكن أن تكون قيمة المتغير أما رقم أو نص أو حرف أو رقم عشري أو....

وإذا أردنا أن ننشئ متغير في ذاكرة الحاسوب فيجب أن نحدد نوع القيمة التي سيجعلها المتغير , مع العلم أننا نختار اسم الكائن كما نشاء مادام الاسم يحقق الشروط التالية:

1- أن يكون اسم المتغير حروف أو أرقام لاتينية

2- أن لا يكون أول حروف الاسم رقم , فهذا الاسم غير مسموح **int 5** بينما هذا مسموح **int5**

3- أن لا يحتوي الاسم على علامات أو إشارات مثل # أو \$ أو * أو + , الإشارة المسموح فقط هي _

وهذه الشروط تنطبق على أسماء الكائنات أو المتغيرات التي سنقوم بإنشائها.

إذا المتغير هو كائن يستخدم لتخزين قيمة من نوع محدد

مثلاً إذا أردنا أن نعرف متغير اسمه x من نوع رقم سيكون الكود كالتالي:

كود

```
int x;
```

وكلمة **int** اختصار لكلمة **integer** ومعناها عدد صحيح , الكود السابق معناه أنا عرفنا متغير اسمه x سيجعل قيمة من نوع عدد صحيح , لاحظ أنك تستطيع تغيير اسم المتغير كما نشاء المهم أن تراعي شروط التسمية الثلاثة , وإذا أردنا تحميل هذا المتغير قيمة محدد .

يكون الكود كالتالي:

كود

```
x=20;
```

الكود السابق معناه أن المتغير x يحمل القيمة 20 ويستمر بحمل القيمة حتى يتوقف البرنامج أو نحمله قيمة أخرى
لاحظ الكود التالي:

كود

```
int Var1;  
int Var2;  
Var1=10;  
Var2=Var1;
```

قمنا بتعريف متغيرين من نوع رقم صحيح ثم حملنا الرقم الأول قيمة 10 , وفي السطر الأخير حملنا المتغير الثاني قيمة المتغير الأول أي أن المتغير الثاني يحمل أيضاً القيمة 10
ملاحظة مهمة: في لغات البرمجة معامل التساوي ليس تبادلي يعني أن

Var2=Var1

لا تساوي أبداً

Var1=Var2

لأن معامل = في لغات البرمجة يقوم بنقل قيمة المتغير في اليمين إلى المتغير في اليسار
فلو عكسنا المعاملات في السطر الأخير من الكود السابق هكذا :

كود

```
int Var1;  
int Var2;  
Var1=10;  
Var1=Var2;
```

فسيظهر لنا خطأ عند السطر الأخير لأن المعالج سيحاول نقل ما في المتغير Var2 إلى المتغير Var1, ولكن المتغير Var2 لا يحتوي على أي قيمة لذلك سيظهر لنا الخطأ .

بعض أنواع المتغيرات وكيفية تعريفها:

int رقم صحيح

string نص أو كلمة

double رقم حقيقي رقم يقبل الكسور العشرية مثل "41.51"

char حرف

Date Time تاريخ أو وقت

bool قيمة بوليا نية

معنى القيمة البوليانية أي أن المتغير يحمل قيمة صح أو خطأ , للتوضيح:

كود

```
int k;  
string w;  
double m;  
char n;  
bool p;  
k=15;  
w="نص هذا";  
m=12.15;  
n='م';  
p=true;
```

لا حظ أن المتغير من نوع رقم لا يحتاج إلى علامتي اقتباس , بينما المتغير من نوع نص أو حرف يحتاج إلى علامتي

اقتباس , المتغير من نوع **bool** يحمل أما قيمة **true** أو **false**

ماذا لو جربنا الكود التالي:

كود

```
k="نص";
```

سيظهر لنا خطأ لأننا نحاول أن نحمل المتغير **k** قيمة من نوع نص , بينما المفروض أن يحمل قيمة من نوع رقم كما عرفناه

وأيضاً هذا الكود خاطئ لنفس السبب:

كود

```
k=w;
```

حيث أن w يحتوي على قيمة من نوع نص و k المفروض أن يحمل قيمة من نوع رقم ,ماذا لو كان لدينا هذا الكود:

كود

```
w="45";
```

```
k=w;
```

سيظهر لنا خطأ أيضاً عند السطر الثاني بالرغم من أن w يحمل رقم ,لأن المترجم لا يعرف أن w يحمل رقم , فهو يعتبر كل ما هو بين علامتي اقتباس على أنه نص ,وبالتالي فالمترجم لن يقبل أن يحمل k قيمة نصية لأن من المفروض أن يحمل رقم ,للتغلب على هذه المشكلة يوجد في بيئة التطوير كائن للتحويل ما بين الأنواع ,وهذا الكائن يقوم بتحويل القيمة إلى أي نوع فإذا أردنا تنفيذ الكود السابق باستخدام كائن التحويل:

كود

```
w="45"; k=Convert.ToInt32(w);
```

الكائن **Convert** قام بأخذ قيمة w وحولها إلى رقم ثم حملها للمتغير ,لاحظ أننا للوصول إلى أفعال أو صفات الكائن نكتب اسم الكائن ثم نقطة ثم فعل أو صفة الكائن ,فالكائن **Convert** يحتوي على فعل **ToInt32** لذلك كتبنا اسم الكائن ثم نقطة ثم فعل الكائن ,ووضعنا بين القوسين المدخلات , ومن أفعال الكائن **Convert**:

ToInt32 للتحويل إلى رقم صحيح كما رأينا في المثال السابق

ToString للتحويل إلى نص

toDouble للتحويل إلى عدد عشري

ToChar للتحويل إلى حرف

ToBoolean للتحويل إلى قيمة بوليانية

ملاحظة: الرقم الأخير من المتغير **ToInt32** يعني حجمه في الذاكرة, مثلاً **int16** حجمه 16 بت و **int32** حجمه 32 بت ويوجد أيضاً 64, وكل نوع له مدى محدد لا يستطيع يتجاوزه, مثلاً **int16** يستطيع خزن أي رقم من - 32,768 إلى

32,767

و **int32** يستطيع خزن أي رقم من - 2,147,483,648 إلى 2,147,483,647

و **int64** له مدى أكبر بكثير, لذلك اخترنا **int32** كحل وسط , لأنه يخزن أرقام كبيرة , ولا يستهلك مكان كبير في الذاكرة



تطبيق الدرس الثاني:

افتح مشروع التطبيق الذي طبقنا عليه في الدرس الماضي غير النص المكتوب في الزر من "ترحيب" إلى "إظهار قيمة x" من خلال تعديل الصفة Text من جدول الصفات , انقر نقراً مزدوجاً على الزر لتنتقل إلى شفرة الزر .. ثم قم بحذف سطر الأوامر التالي:

كود

```
MessageBox.Show("مرحباً بكم إلى سي شارب");
```

في مكان الكود المحذوف أكتب الكود التالي:

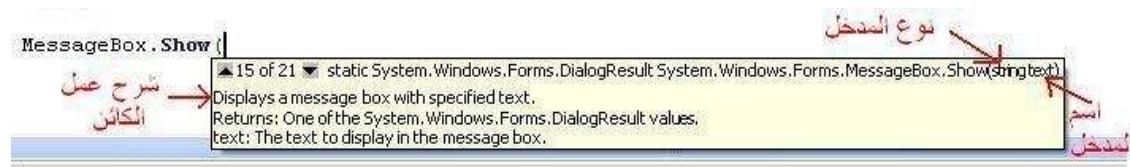
كود

```
int x;
x=50;
MessageBox.Show(x);
```

في السطر الأول عرفنا متغير اسمه x سيحمل قيمة من نوع رقم صحيح , في السطر الثاني حملنا المتغير x قيمة 50 في السطر الثالث كتبنا أمر اظهار الرسالة , لاحظ أننا وضعنا x بدل نص الرسالة , حيث أن المترجم سيقوم بإظهار رسالة ويكتب داخلها القيمة المحملة على x , قبل تجربة البرنامج , هل تعتقد أن هناك خطأ في الشفرة السابقة؟

نعم هناك خطأ فلو جربت تشغيل البرنامج ثم الضغط على الزر ستظهر لك رسالة خطأ , وسبب الخطأ أن كائن `MessageBox.Show` يحتاج إلى مدخلات من نوع نص ليظهرها في الرسالة , وقد مررنا x على أنه المدخلات ولكنه من نوع رقم صحيح

ربما يتساءل البعض ما أدراني أن الكائن `MessageBox.Show` يحتاج إلى مدخلات من نوع نص وليس من نوع رقم الأمر بسيط جداً , إذا كنت تريد ان تعرف نوع المدخلات التي يحتاجها الكائن سيظهر لك ذلك في رسالة صغيرة بمجرد أن تفتح القوس الأول كما في الصورة:



عندما انتهيت من كتابة القوس الأول ظهر صندوق المعلومات وأظهر أسماء وأنواع المدخلات التي يحتاجها الكائن السهم يشير إلى الكلمة التالية:

إقتباس

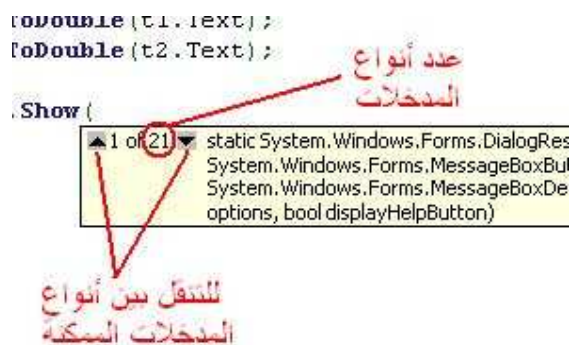
string text

الجزء الأول من الكلمة السابقة يعني نوع المدخل وهو string أي نص , والجزء الثاني هو أسم المدخل text وهذا سنشرحه فيما بعد , ولكن ما بهمنا هو الجزء الأول لنعرف نوع المدخل الذي يحتاجه الكائن .

ملاحظة: بعض الكائنات ومنها كائن **MessageBox.Show** يمكن أن تقبل مدخلات متعددة فيمكن أن تقبل مدخل واحد وهو نص الرسالة كما رأينا في الدرس السابق و يمكن أن تقبل أكثر من مدخل .

الكائن **MessageBox.Show** يقبل 21 نوع من المدخلات

ومن الممكن أن تتجول بين أنواع المدخلات الممكنة للكائن بالضغط على زر الأسهم فوق وتحت في لوحة المفاتيح عند ظهور صندوق معلومات المدخلات كما في الصورة:



ولكن ما بهمنا حالياً من أنواع المدخلات هو النوع رقم 15 الذي يحتاج فقط لنص الرسالة , نعود إلى الكود السابق الذي أظهر لنا خطأ نتيجة تضارب أنواع البيانات .

لحل مشكلة التضارب نستخدم كائن التحويل لتحويل قيمة x إلى نص ثم تمريرها إلى الكائن

MessageBox.Show

لأنه كما قلنا لا يقبل إلا مدخلات من نوع نص و x من نوع رقم , الكائن الذي يحول من رقم إلى نص هو

Convert.ToString

وبالتالي سيصبح الكود الصحيح هكذا:

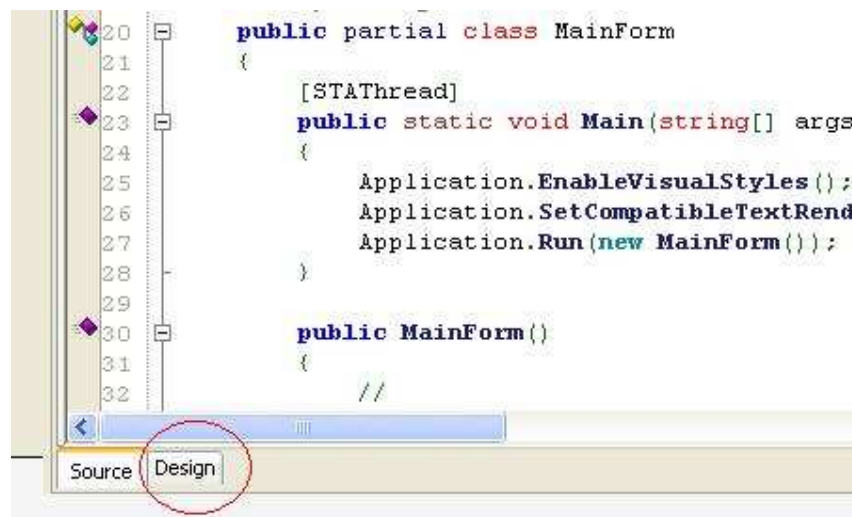
كود

```
int x;  
x=50;  
MessageBox.Show(Convert.ToString(x));
```

جرب البرنامج الآن واضغط على الزر ستظهر لك رسالة مكتوب عليها 50 ، وهكذا تغلبنا على مشكلة تضارب البيانات



لتوسيع التطبيق سنستخدم كائن آخر من كائنات البيئة وهو مربع النص ، اضغط على زر **Design** أسفل الشفرة لترجع إلى نافذة التصميم :



الآن قم بسحب كائن مربع النص وأفلته فوق النافذة كما سحبت الزر في الدرس الأول



ليصبح شكل النافذة هكذا:



تأكد من اختيار مربع النص واذهب إلى جدول الصفات، أبحث عن الصفة (Name) ستجد مكتوباً أمامها **textBox1** غيرها إلى **t1**.

لاحظ أنا غيرنا أسم الكائن من **textBox1** إلى **t1** وذلك لسهولة التعامل معه في الشفرة كما سنرى لاحقاً. الآن اضغط مزدوجاً على الزر لترجع إلى الشفرة، أمسح السطر الثاني من الكود السابق.

وأكتب بدل منه هذا الكود

كود

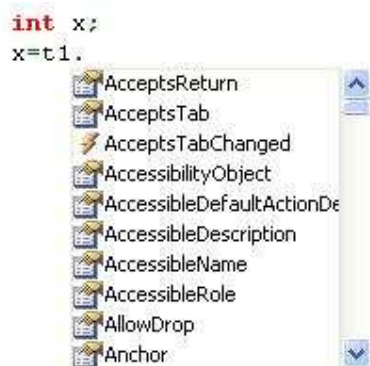
```
x=t1.Text;
```

لاحظ أن الكائن **t1** الذي هو مربع النص له صفة اسمها **Text** وهي تمثل النص المكتوب عليه ,معنى الكود السابق أن المترجم سيقوم بأخذ ما هو مكتوب في مربع النص المسمى **t1** ويحمله للمتغير **x** , أي أن الكود كاملاً سيصبح هكذا:

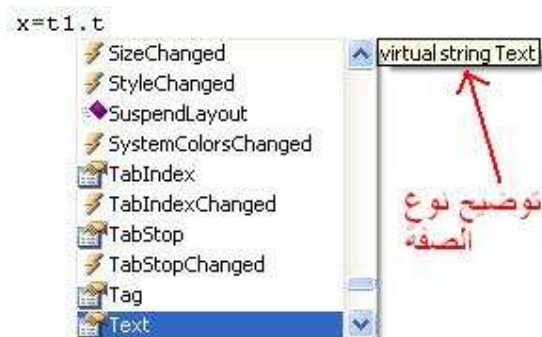
كود

```
int x;
x=t1.Text();
MessageBox.Show(Convert.ToString(x));
```

ملاحظة: لمعرفة صفات وأفعال الكائن أكتب أسم الكائن ثم أكتب نقطة , ستظهر لك نافذة بجميع أعمال و صفات الكائن ففي المثال السابق بمجرد كتابة تكتب **t1** ثم نقطة , سيظهر لك هذا الصندوق :



وبمجرد أن تكتب حرف **t** من كلمة **Text** سينقل الاختيار في الصندوق إلى كلمة **Text**:



وللاختصار اضغط على زر Tab لكي تكتب كلمة Text آلياً

لاحظ في الصورة السابقة أن نوع صفة Text هو string أي نص, وبالتالي لو حاولت تنفيذ البرنامج وضغط الزر سيظهر خطأ لأننا حاولنا أن نحمل المتغير x قيمة نصية, بينما يجب أن نحمله رقم, يأتي دور كائن التحويل مرة أخرى ولكن هذه المرة من نص إلى رقم, فيصبح الكود الكامل والصحيح هكذا:

كود

```
int x;  
x=Convert.ToInt32(t1.Text);  
MessageBox.Show(Convert.ToString(x));
```

في السطر الأول عرفنا متغير من نوع رقم, وفي السطر الثاني حملنا هذا المتغير القيمة المكتوبة في مربع النص وحولناها إلى رقم, وفي السطر الثالث لم نغير شيء.
الآن نفذ البرنامج, وقبل أن تضغط الزر أكتب أي رقم في مربع النص ثم اضغط الزر, ستظهر لك رسالة بالرقم الذي كتبتة تأكد أنك ستكتب رقم وليس حرف وإلا ظهرت لك رسالة خطأ, لتوسيع التطبيق أضف متغيرين آخرين y,z وأضف مربع نص آخر وسمه t2, غير النص المكتوب على الزر وأكتب بدلاً منه "اظهار ناتج الجمع".
ليكون شكل النافذة هكذا:



الآن اذهب إلى الشفرة وعدلها إلى التالي:

كود

```
int x;  
int y;  
int z;  
x=Convert.ToInt32(t1.Text);  
y=Convert.ToInt32(t2.Text);  
z=x+y;  
MessageBox.Show(Convert.ToString(z));
```

عرفنا ثلاثة متغيرات من نوع رقم صحيح , في السطر الرابع حملنا x الرقم المكتوب في $t1$, وفي السطر الخامس حملنا y الرقم المكتوب في $t2$, وفي السطر السادس حملنا z مجموع x و y , وفي السطر السابع أظهرنا قيمة z في الرسالة , لاحظ أنا وضعنا z بدل x في السطر الأخير , الآن قم بتنفيذ البرنامج , وأكتب عدداً صحيحاً في كل مربع نص واضغط الزر. ستظهر لك نافذة مجموع العددين :



اغلق الرسالة وجرب أن تضع في أحد مربعات النص رقماً حقيقياً مثل 41.25 واضغط على الزر , ستظهر لك رسالة خطأ لأن المتغيرات z و y و x تحمل متغيرات من نوع رقم صحيح وأنت حاولت أن تحملها رقم حقيقي , اضغط على زر Continue في رسالة الخطأ

ليتم إيقاف تنفيذ البرنامج:



أذهب إلى الشفرة وحاول أن تجد الحل للقيام بجمع أعداد حقيقية, وذلك هو الواجب.

تلميح: نعرف أن **double** تعني رقم حقيقي والكائن **Convert.ToDouble** يحول إلى رقم حقيقي, ما عليك إلا أن

تضع **double** بدل من **int**, وتضع **Convert.ToDouble** بدل من **Convert.ToInt32**

لحظة, لم ينتهي الواجب بعد, أضف إلى البرنامج ثلاثة أزرار إضافية للقيام بباقي العمليات الرياضية

1- اظهر ناتج الطرح

2- اظهر ناتج الضرب

3- اظهر ناتج القسمة

ليصبح شكل النافذة هكذا:



نهاية الدرس الثاني.

