Visual Basic . Net

المصفوفات البرامترية
Parameter Arrays

الشرح الوافي و الكافي

إعـــــداد

رزقي توفيق

باتـــنة - الجزائـــر

2012

/http://www.vb4arab.com/vb

المصفوفات البرامترية
Parameter Arrays

قبل البدء : تمهيد

قبل البـدء يستوجب مني قبل الخوض فـــي التفاصيل التطرق لشرح بعض المفاهيم الأساسية لتقريب وتنسيق الأفكار بيني و بين القارئ المتتبع لهذا الموضوع .
كما هو معلوم أن الإجراءات و الدوال يمكن أن تحوي بين الأقواس قائمة من البرامترات Parameters ذات الأنواع المختلفة :

```
Sub Subname( Parameterlist )
```

```
Function Functionname( Parameterlist ) As ReturnType
```

الفرق بين الإجراء Sub و الدالة Function :

الفرق بينهمـــا أن الإجراء Sub لا يعود بقيمة ، بينما الدالة Function قادرة على العودة بقيمة .

ستنصب شروحاتنا هذه على الإجراءات وكل ما يقال في الإجراءات ينطبق على الدوال .

```
Sub MySub(Parameter 1, Parameter 2, Parameter 3, ..... , Parameter n)
```

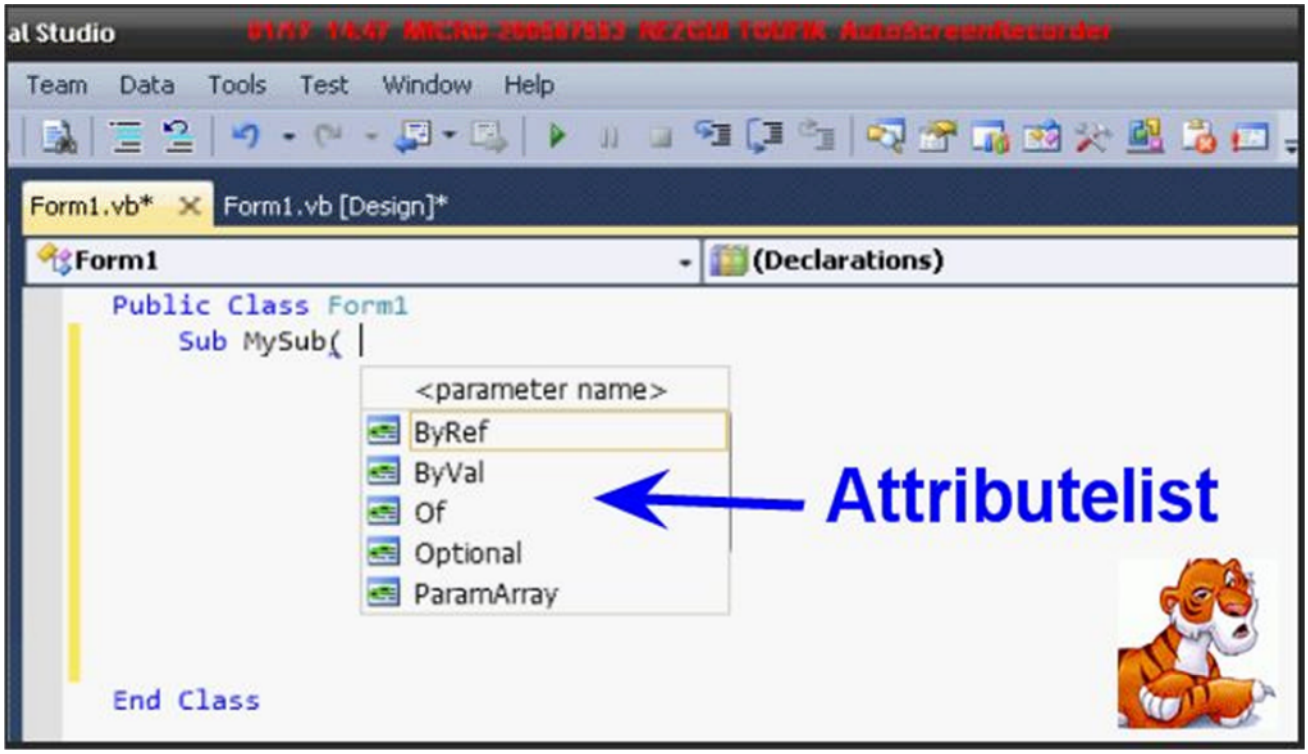
ويأخذ البرامتر الواحد الصيغة التالية :

```
[<AttributeList>] ParameterName[( )] [As ParameterType] [= DefaultValue]
```

```
ByVal Name As String ="rezguicondor"
```

AttributeList = قائمة الصفات

وهي تمثل الصفات التي سيتم تطبيقها على العنصر ، وهي تحدد عموما الطريقة التي يتم بها تمرير القيم عند إستدعاء الإجراءات و الدوال و الخصائص .



ParameterName = إسم البرامتر

إسم المتغير المحلي الذي يمثل البرامتر

ParameterType = نوع البرامتر

نوع بيانات المتغير المحلي الذي يمثل البرامتر و قد يكون عدد صحيح Integer أو سلسلة نصية String أو بايت Byte بالإضافة إلى الأنواع الأخرى المعروفة .

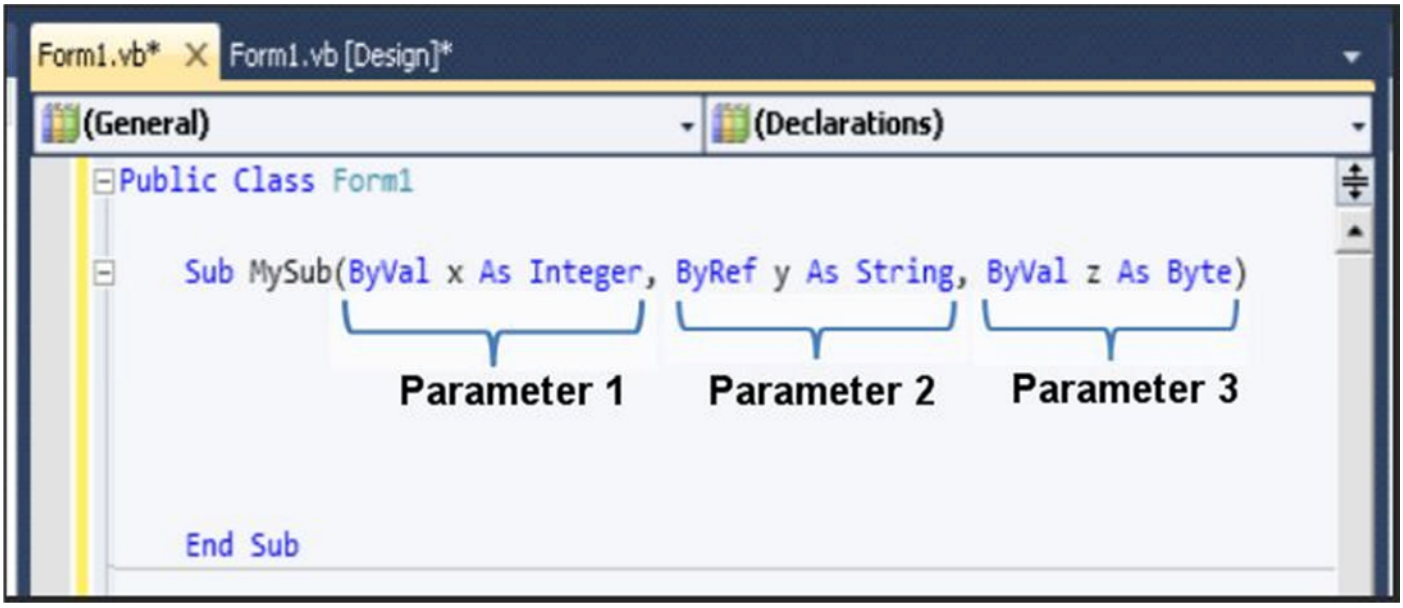
DefaultValue = القيمة الإفتراضية

مطلوبة للبرامترات الإختيارية Optional Parameters ، تمثل بثابت أو عبارة ثابتة تقيم نوع بيانات البرامتر . فإذا كان النوع عبارة عن كائن أو فئة أو واجهة أو مصفوفة أو تركيبة فإن القيمة الإفتراضية يمكن أن تكون فقط لا شيء Nothing .

البرامترالإختياري هو كل برامتر يتم تسبيقه بالكلمة المحجوزة Optional . ويشترط تعيين قيمة إفتراضية له عند تعريف الإجراء .

```
Sub MySub(Optional src As Integer = 1)
Sub MySub(Optional src As String = "rezguicondor")
```

بعد أن أخذنا ولو فكرة عن برامترات الإجراء .



كما هو مألوف أنه عند القيام بتعريف اجراء او دالة عادة نقوم بتحديد قائمة البرامترات للاجراء او الدالة . وعند إستدعاء هذا الإجراء لا يمكن ارسال اكثر من القيم المطلوبة على حسب عدد البرامترات المعرفة في الإجراء . ولكن ماذا لو اردنا ارسال عدد غير محدود من القيم دون التقيد بعدد برامترات الإجراء ،
عندها سوف نكون بحاجة ماسة إلى وسيلة برمجية قد تكون دالة أو إجراء للتعامل مع مجموعة غير محدودة من العناصر و التي قد تكون أعداد صحيحة أو عشرية أو سلاسل نصية أو بايتات أو نوع آخر من أنواع البيانات مع الحرية في تمرير العدد الغير محدود من العناصر المختلفة القيم في كل مرة .

وهنا يأتي دور المصفوفات البرامترية Parameter Arrays والتي هي محور شروحاتنا هذه .

```
Sub MySub(Parameter 1, Parameter 2, Parameter Array)
```

و يأخذ برامتر المصفوفة البرامترية Parameter Array الصيغة التالية :

```
ByVal ParamArray Array_name( ) As type
```

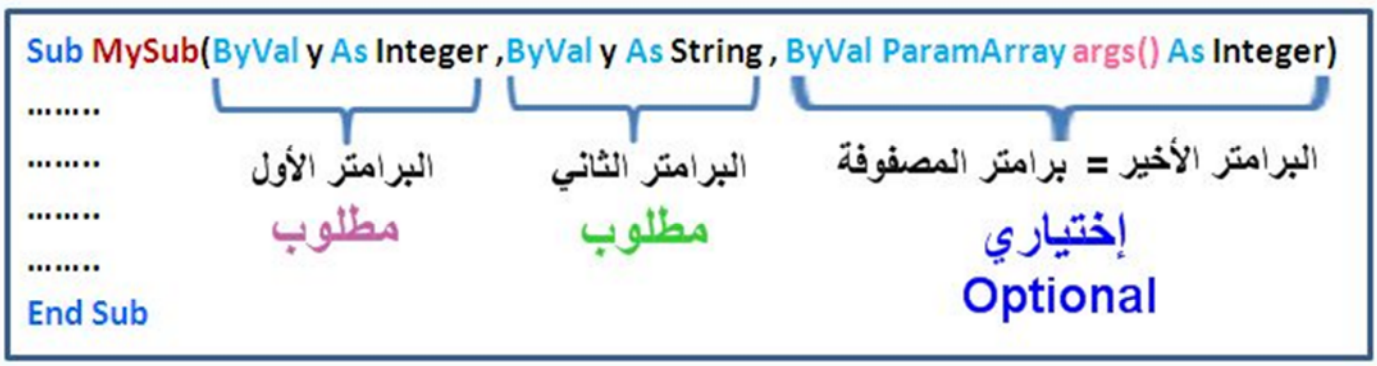
ParamArray = يجب تضمين هذه الكلمة الدليلية ParamArray للدلالة على المصفوفة البرامترية Parameter Array في قائمة البرامترات Parameter List .
Array_name = إسم المصفوفة البرامترية .
Type = نوع بيانات برامتر المصفوفة .

وهناك بعض الشروط الواجب توفرها لإستخدام Parameter Array :

يجب تضمين الكلمة الدليلية ParamArray للدلالة على المصفوفة البرامترية Parameter Array في قائمة البرامترات Parameter List ، مع تطبيق القواعد التالية :

- يمكن للإجراء تعريف برامتر مصفوفة واحد و واحد فقط ، و الفيجوال بيسك نفسه لا يسمح بإدراج برمترات مصفوفات أخرى بعد برامتر المصفوفة الأولى ، المهم أنه لا يمكن تعريف اكثر من مصفوفة برامترية في نفس الاجراء .
- و يجب أن يكون برامتر المصفوفة هو البرامتر الأخير في قائمة برمترات الإجراء عند تعريفه .

- يجب أن يتم تمرير المصفوفة البرامترية بالقيمة By Value .
- المصفوفة البرامترية هي إختيارية تلقائيا Automatically Optional بدون تسبيقها بالكلمة Optional ، و قيمتها الافتراضية هي مصفوفة فارغة أحادية البعد Empty One-Dimensional Array من نوع عنصر المصفوفة البرامترية .
- كل البرامترات التي تسبق المصفوفة البرامترية يجب أن تكون مطلوبة أي غير إختيارية . و المصفوفة البرامترية يجب أن تكون البرامتر الإختياري الوحيد Only Optional Parameter .



عليك أن تكون ملما بهذه الشروط و حافظا للصيغة الخاصة بتعريف برامتر المصفوفة البرامترية و سيتولى الفيجوال بيسك مهمة إظهار الأخطاء و تذكيرك بالشروط الواجب إتباعها لتداركها .

فالإجراء التالي :

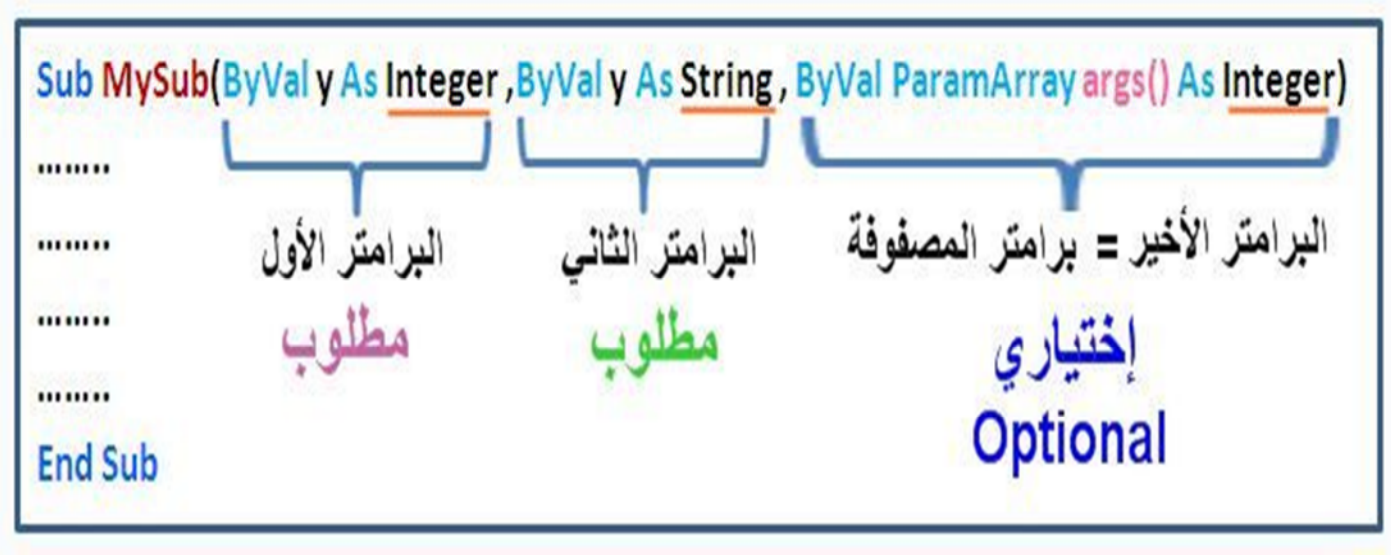
```
Sub MySub(ByVal y As String , ByVal ParamArray args() As Integer)
```

يمكن تصوره بهذا الشكل :

```
Sub MySub(ByVal y As String , ByVal args1 As Integer, ByVal args2 As Integer,
ByVal args3 As Integer, ..... , ByVal argsn As Integer)
```

هذا فيما يخص الأمور الواجب معرفتها عند تعريف الإجراء الذي يحتوي على مصفوفة براميتريه ، أما عند إستدعاء هذا النوع من الإجراءات فهناك نقاط يجب توضيحها :

لنفترض أننا قمنا بتعريف الإجراء التالي :



كما نلاحظ أن هذا الإجراء يحتوي على قائمة من ثلاثة برمترات :

- البرامتر الأول هو برامتر عادي مطلوب من نوع الأعداد الصحيحة Integer .
- البرامتر الثاني هو برامتر عادي مطلوب من نوع السلاسل النصية String .
- البرامتر الثالث و الأخير هو برامتر مصفوفة وهو إختياري من نوع الأعداد الصحيحة Integer

كما هو معلوم يتم إستدعاء الإجراء من أي مكان بإستخدام الكلمة المحجوزة Call أو بالإستغناء عنها وهو ما يسمح به الفيجوال بيسك .

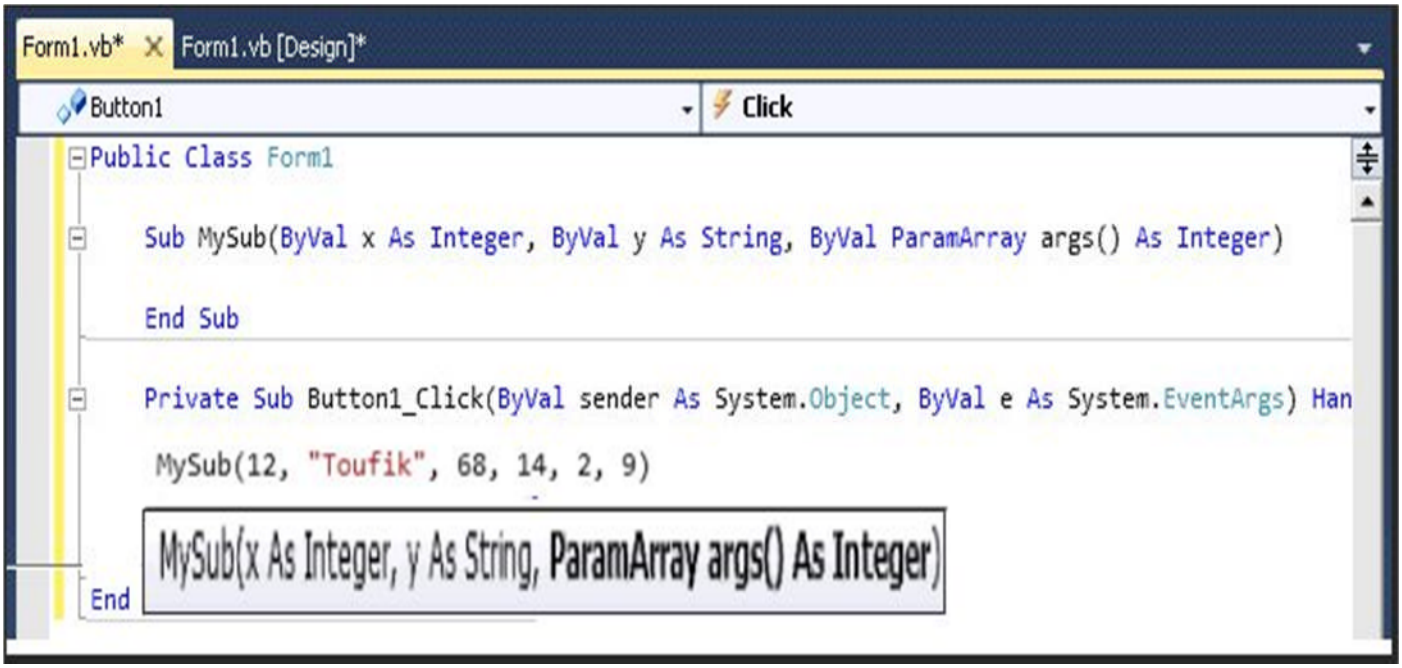
```
Call MySub(12, "Toufik", 68, 14, 2, 9)
MySub(12, "Toufik", 68, 14, 2, 9)
```

ويطلق على القيم المدخلة بين قوسي الإجراء عند إستدعائه بالوسيطات Arguments و التي تقابل البرامترات Parameters في تعريف الإجراء .

```
Call MySub(Argument 1, Argument 2, Argument 3, ..... , Argument n)
```

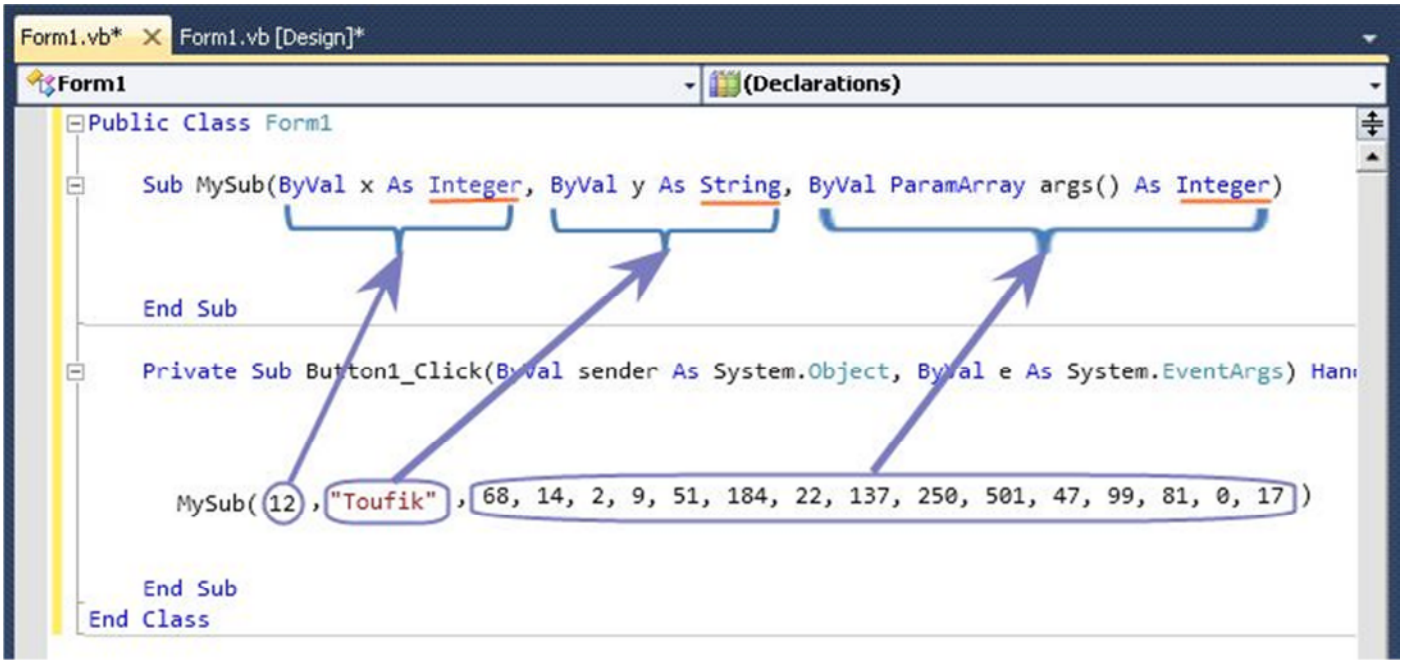
و يطلق على مجموعة الوسيطات بين قوسي الإجراء عند إستدعائه إسم قائمة الوسيطات Argument list .

```
Call MySub( Argumentlist )
```



كما تلاحظ أنه عند إستدعاء الإجراء السابق الذكر قمنا :

- بإدخال العدد الصحيح 12 ومطلوب إدخاله إجباريا كوسيطة أولى لتقابل البرامتر الأول المعرف في الإجراء والذي هو من نوع الأعداد الصحيحة Integer .
- كما قمنا بإدخال السلسلة النصية "Toufik" ومطلوبا إدخالها إجباريا كوسيطة ثانية لتقابل البرامتر الثاني المعرف في الإجراء والذي هو من نوع السلاسل النصية String .
- وفي الأخير قمنا بإدخال سلسلة من الأعداد الصحيحة 9 ,2 ,14 ,68 و هي إختيارية حيث يمكننا إدخال عدد غير محدود من الأعداد الصحيحة كوسائط متتابعة مفصولة بفاصلة لتقابل جميعها البرامتر الأخير و هو برامتر المصفوفة المعرف في الإجراء والذي هو من نوع الأعداد الصحيحة Integer كما يمكننا عدم إدخال أي قيمة مكان الوسيطات المقابلة لبرامتر المصفوفة أي إهمالها و الإستغناء عنها نهائيا .



كل هذه الإستدعاءات الخاصة بالإجراء السابق صحيحة :

```
Call MySub (5, "Rezgui", 10, 15, 20, 25)
Call MySub (17, "Ramzi")
Call MySub (245, "Riadh", 0)
Call MySub (5, "336", 10, 15, 20, 25)
Call MySub (43, "25",110, 135, 20, 25 , 51 , 84 , 272 , 371)
Call MySub (75, "Raouf",110, 135, 20, 25 , 51 , 84 , 272 , 371, 8 , 16)
```

كما تلاحظ في الإستدعاء الثاني أننا قمنا بالإستغناء عن الوسيطات المقابلة لبرامتر المصفوفة وسيعيد الإجراء القيمة صفر في هذه الحالة .

كل هذه الإستدعاءات الخاصة بالإجراء السابق خاطئة :

```
Call MySub (5)
Call MySub ("Ramzi", 78)
Call MySub ("Ramzi", "Riadh")
Call MySub (245, "Riadh", "Raouf", 15, 20, 25)
Call MySub ( "336", 5, 10, 15, 20, 25)
```

خلاصة القول عند إستدعاء الإجراء يتوجب علينا :

- إحترام الترتيبة البرامترية كما هي معرفة في تصريح الإجراء .
- إدخال القيم إجباريا للوسائط المطلوبة أي الغير إختيارية إذا ما تم تضمينها في تصريح الإجراء ولا يمكن إهمالها و الإستغناء عنها على عكس برامتر المصفوفة .
- التقيد بإدخال أنواع البيانات الملائمة بناء على أنواع بيانات البرمترات المقابلة لها في تصريح الإجراء .

وكل ما قيل في الإجراءات ينطبق على الدوال ، وما عليك سوى إستغلال ما في جعبتك من مهارات برمجية لكتابة الأكواد الجيدة و الملائمة لإجراءاتك و دوالك وإحتراف إستخدام :

- جمل الشرط If ...The ...Enf If
- الحلقات التكرارية

For ... Next

For Each ... Next

- دالتي (الطريقتان)

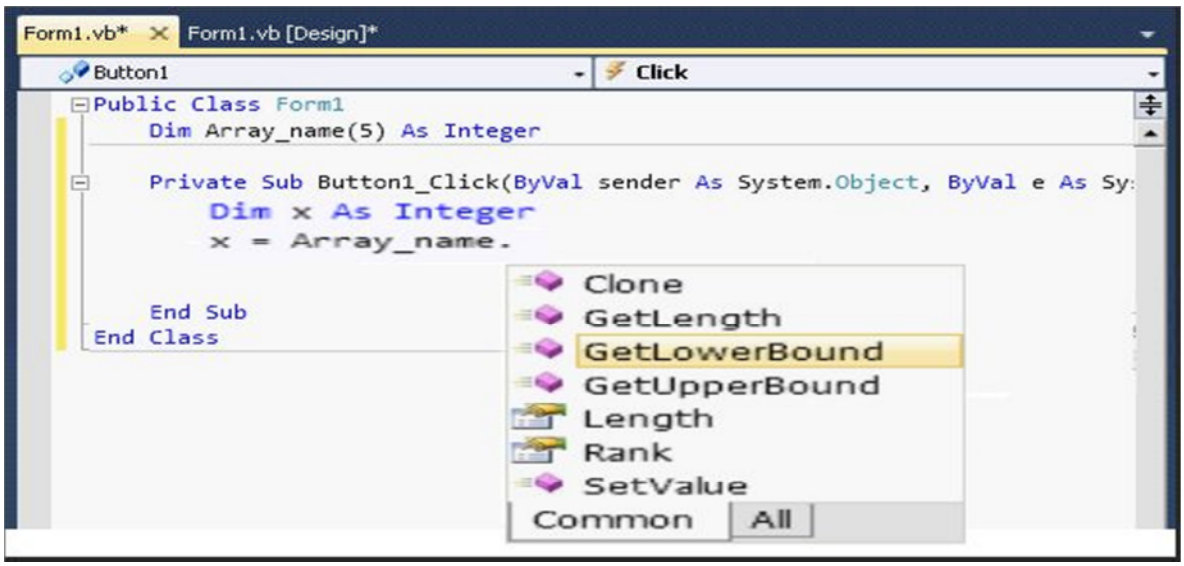
LBound(Array_name) و UBound(Array_name)

أو

Array_name.GetLowerBound(0) و Array_name.GetUpperBound(0)

اللتان تعودان على التوالي بالعنصر الاول والاخير للمصفوفة .

بالإضافة إلى طرق و خصائص المصفوفات الأخرى :



وكل هذا يدخل ضمن تلك المهارات للتعامل مع قيم المصفوفة البرامترية على أكمل وجه .

الفرق بين الـ Sub و الـ Function :

الفرق بينهمـــا أن Sub لا يعود بقيمة ، بينما الـ Function قادرة على العودة بقيمة .

```
Sub Subname( Parameterlist )
```

```
Functionname( Parameterlist ) As ReturnType
```

إستخدام Exit Function دون إسناد قيمة للدالة فستعود بقيمة 0 إن كانت عددية Integer ، لاشى "" إن كانت حرفية String ، أو Nothing إن كانت Object .

المصفوفات البرامترية
Parameter Arrays

تعريف المصفوفات البرامترية

عادة ، نحن لا يمكننا إستدعاء إجراء يحتوي على عدد من الوسيطات Arguments أكثر من العدد الذي يتم تحديده أثناء التصريح بالإجراء ، عندما نكون في حاجة الى عدد غير محدد من الوسيطات ، يمكننا التصريح بالمصفوفة البرامترية Parameter Array . والتي تسمح للإجراء بقبول مصفوفة من القيم للبرامتر . و ليس من الضروري علينا معرفة عدد العناصر في المصفوفة البرامترية عند تعريف الإجراء . ويتم تحديد حجم المصفوفة عند كل إستدعاء للإجراء .

Declaring a ParamArray
التصريح بـ ParamArray

يمكننا إستخدام الكلمة الدليلية ParamArray للدلالة على المصفوفة البرامترية Parameter Array في قائمة البرامترات Parameter List . مع تطبيق القواعد التالية :

- يمكن للإجراء تعريف مصفوفة برامتر واحدة فقط ، و يجب أن تكون البرامتر الأخير في تعريف الإجراء .
- يجب أن يتم تمرير المصفوفة البرامترية بالقيمة By Value . و إنه لأمر جيد لممارسة البرمجة تضمين الكلمة الدليلية ByVal في تعريف الإجراء .
- المصفوفة البرامترية هي إختيارية تلقائيا Automatically Optional . قيمتها الافتراضية هي مصفوفة فارغة أحادية البعد Empty One-Dimensional Array من نوع عنصر المصفوفة البرامترية .

- كل البرامترات التي تسبق المصفوفة البرامترية يجب أن يتم طلبها أي غير إختيارية . و المصفوفة البرامترية يجب أن تكون البرامتر الإختياري الوحيد Only Optional Parameter .

Calling a ParamArray
ParamArray استدعاء

مثال :

المثال التالي يقوم بتحديد و إستدعاء الدالة calcSum . و الكلمة الدليلية Paramarray للمصفوفة البرامترية args() تعمل على تمكين الدالة لقبول عدد متغير من الوسيطات .

قم بإضافة زر إلى النموذج Form1 ، أنسخ الكود التالي في نافذة الكود للنموذج ، ثم قم بتنفيذ التطبيق و أنقر على الزر لترى النتيجة .
و الدالة calcSum تقوم بحساب و إرجاع مجموع قيم عناصر الوسيطات التي نحددها عند إستدعاء هذه الدالة من خلال الرسالة MsgBox بالنقر على الزر .

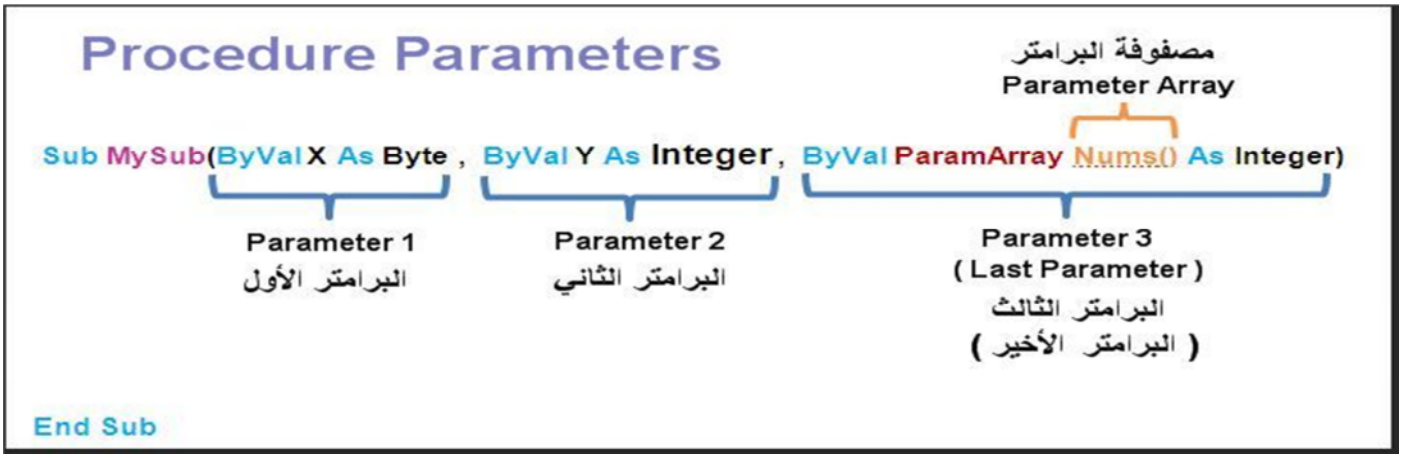
2+3+4+5+6 = 20

قم بتغيير عناصر الوسيطات و لاحظ التنيجة في كل مرة .

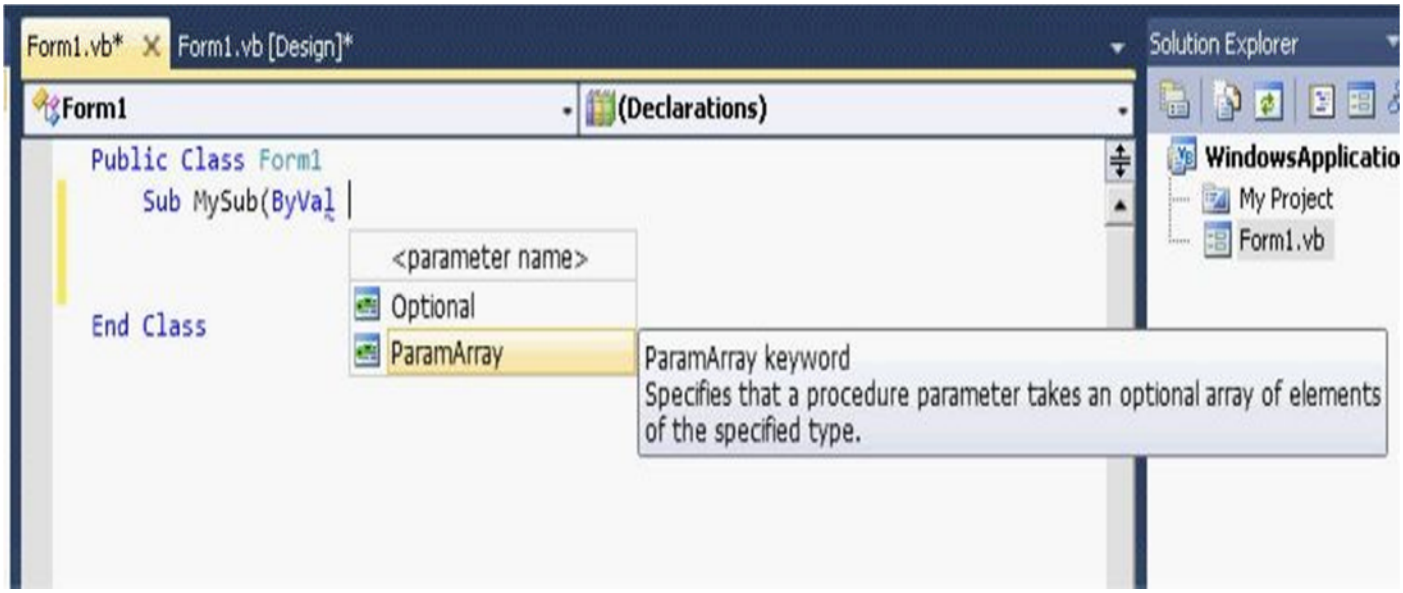
```
Public Class Form1

    Public Function calcSum(ByVal ParamArray args() As Double) As Double

        calcSum = 0
        If args.Length <= 0 Then Exit Function
        For i As Integer = 0 To UBound(args, 1)
         ' calcSum = calcSum + args(i)
            calcSum += args(i)
        Next i

    End Function

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As _
    System.EventArgs) Handles Button1.Click

        MsgBox("Sum  =  " & CStr(calcSum(2, 3, 4, 5, 6)))

    End Sub


End Class
```

الكلمة الدليلية ParamArray

تحدد بأن برامتر الإجراء Procedure Parameter يأخذ مصفوفة إختيارية Optional Array من عناصر النوع المحدد . ولا يمكن استخدام ParamArray إلا في البرامتر الأخير Last Parameter من قائمة البرامترات Parameter List ، إذ لا يمكن لها أن تسبق برامتر عادي .



يسمح لنا البرامتر ParamArray بتمرير عدد غير محدود من الوسيطات Arguments إلى الإجراء . ويستوجب التصريح به إستخدام الكلمة المحجوزة ByVal أي التمرير بالقيمة دائما ودوما .



يمكننا توفير واحد أو أكثر من الوسيطات Arguments إلى البرامتر ParamArray عن طريق تمرير مصفوفة من نوع البيانات المناسبة و الملائمة . قائمة من القيم المفصولة بفاصلة ، أو لا شيء Nothing على الإطلاق .

ملاحظة أمنية :

حينما نتعامل مع مصفوفة و التي يمكن أن تكون كبيرة بشكل غير محدد ، هناك خطر من اجتياح بعض القدرات الداخلية لتطبيقنا . إذا كنا نقبل المصفوفة البرامترية من الكود الذي يستدعيها Calling Code ، فيجب اختبار طولها واتخاذ الخطوات المناسبة إذا كانت كبيرة للغاية بالنسبة لتطبيقنا .

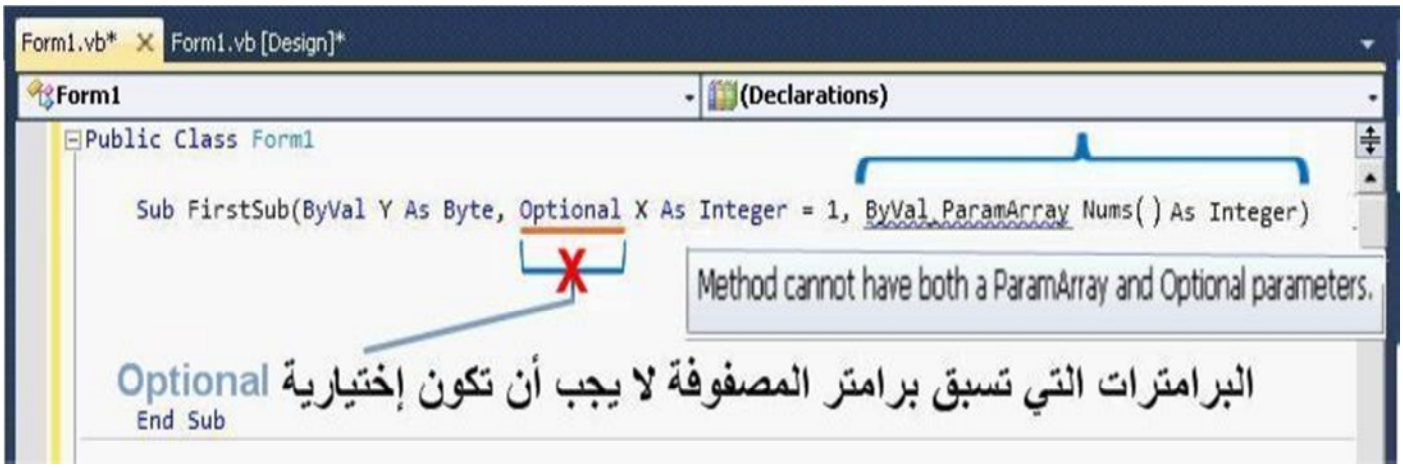
How to: Define a Procedure with an Indefinite Number of Parameters
كيفية تحديد إجراء مع عدد غير محدد من البرامترات

يمكننا التصريح بالمصفوفة البرامترية Parameter Array كما الإدخال الأخير Last Entry في قائمة برامترات الإجراء Procedure's Parameter List .، وهذا يسمح للإجراء من قبول مجموعة من القيم لذلك البرامتر ، بدلا من قيمة واحدة فقط . و ليس من الضروري علينا معرفة عدد القيم في المجموعة عند تعريف الإجراء فالمجموعة يتم تحديدها عند كل استدعاء للإجراء ، و يمكن لكل استدعاء تمرير عدد مختلف من القيم .

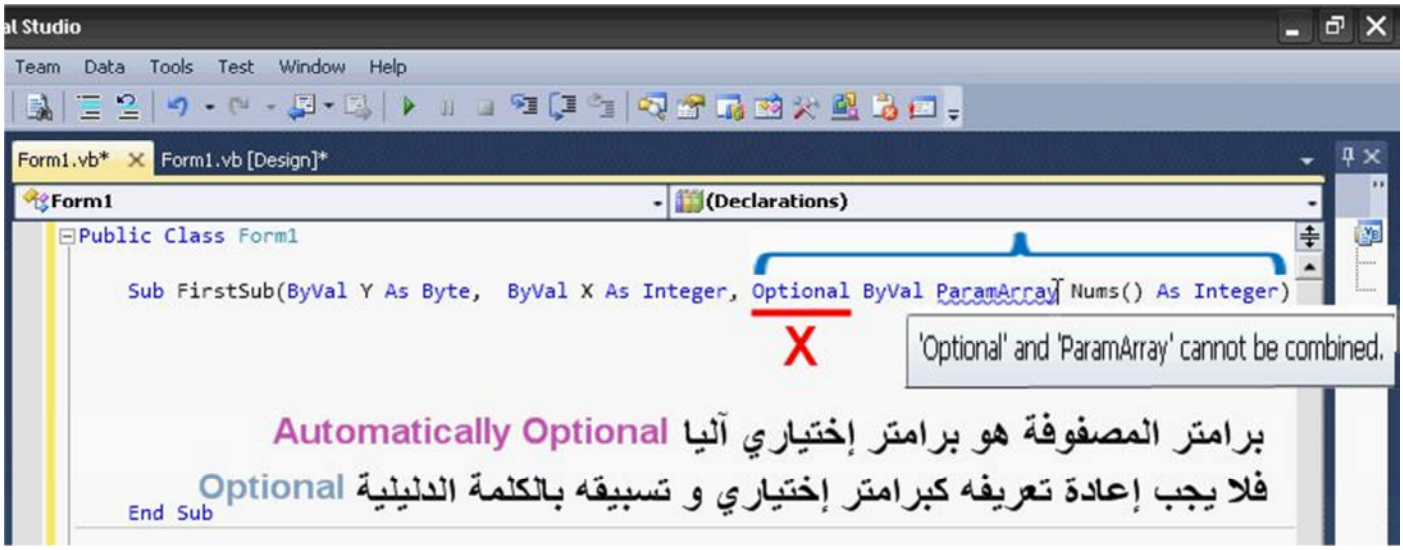
To define a procedure that can accept an indefinite number of values for its last parameter

لتعريف الإجراء الذي يمكن أن يقبل عدد غير محدد من القيم لبرامتره الأخير:

- **في تصريح الإجراء ، تعرف قائمة البرامترات في شكلها الطبيعي ، كل البرامترات ما عدا الأخير يجب أن يتم طلبها أي ليست إختيارية Not Optional .**



- **تسبيق إسم البرامتر الأخير بالكلمات الدليلية ByVal ParamArray . هذا البرامتر هو إختياري تلقائيا Automatically Optional ، و لا يتضمن الكلمة الدليلية Optional .**



- **ويتبع إسم المصفوفة البرامترية زوج فارغ من الأقواس .**
- **و يتبع الأقواس الفارغة الشرط المألوف As .**

- عدم إتباع الشرط As بقيمة إفتراضية Default Value ، القيمة الإفتراضية للمصفوفة البرامترية هي بصورة ألية مصفوفة فارغة أحادية البعد Empty One-Dimensional Array من نوع البيانات الذي نحدده في الشرط As .

Working with the Parameter Array Values
العمل مع قيم المصفوفة البرامترية

الكود ضمن الإجراء يجب أن يعالج المصفوفة البرامترية Parameter Array كمصفوفة أحادية البعد . كل عنصر من العناصر له نفس نوع بيانات المصفوفة البرامترية .
و لمعاجة كل عناصر المصفوفة التي يتم تمريرها إلى المصفوفة البرامترية ، يتوجب على كود الإجراء إستخدام الحلقة التكرارية For ... Next مع تضمين الدالة UBound للحصول على الحد الأعلى للمصفوفة فهي تعيد لنا العدد الإجمالي لعناصر المصفوفة مع اعتبار أن العد يبدأ من الصفر أي أن عنصر المصفوفة الأول يحمل الرتبة صفر و هذا منطق المصفوفات . و بالتالي فإن الحد الأدنى للمصفوفة هو الصفر 0 .

مثال :

يقوم المثال التالي بتعريف إجراء مع المصفوفة البرامترية و ينتج القيم لكل عناصر المصفوفة التي يتم تمريرها إلى المصفوفة البرامترية .

```
Sub studentScores(ByVal name As String, ByVal ParamArray scores() As String)

    MsgBox ("Scores for " & name & ":" & vbCrLf)
  ' Use UBound to determine largest subscript of the array.
  For i As Integer = 0 To UBound(scores)
    MsgBox ("Score " & i & ": " & scores(i))
  Next i

End Sub
```

الأمثلة التالية توضح الإستدعاءات النموذجية للإجراء studentScores :

```
Call studentScores("Anne", "10", "26", "32", "15", "22", "24", "16")
```

```
Call studentScores("Mary", "High", "Low", "Average", "High")
```

```
Dim JohnScores() As String = {"35", "Absent", "21", "30"}
Call studentScores("John", JohnScores)
```

Compiling the Code
ترجمة الكود

يجب التأكد من أن البرامتر **ParamArray** هو الأخير في قائمة البرامترات ، و لا برامتر من البرامترات السابقة يتم التصريح به بصورة إختيارية **Optional** .

How to: Call a Procedure that Takes an Indefinite Number of Parameters
كيفية إستدعاء الإجراء الذي يأخذ عدد غير محدد من البرامترات

يمكن للإجراء التصريح بالإدخال الأخير **Last Entry** في قائمة برامتراته التي ستكون المصفوفة البرامترية **Parameter Array** . و التي تسمح له بقبول عدد غير محدد من القيم لذلك البرامتر، و بدلا من قيمة واحدة فقط .

To call a procedure with a parameter array and omit the corresponding argument
لاستدعاء إجراء مع المصفوفة البرامترية وإهمال الوسيطة المقابلة

```
Call studentScores("George")

Call studentScores(Nothing)
```

- كتابة إستدعاء الإجراء بالشكل الطبيعي . مع وجوب أن تكون المصفوفة البرامترية الوسيطة الأخيرة **Last Argument** .
- إنهاء قائمة الوسيطة تبعا لـ : الوسيطة التالية إلى الأخيرة **Next-to-Last Argument** . المصفوفة البرامترية هي اختيارية **Optional** ، وكل البرامترات السابقة يجب أن يتم طلبها أي غير إختيارية .

أو

الإستعانة بالكلمة المحجوزة Nothing كوسيطة للمصفوفة البرامترية .

- الفيجوال بيسك يمرر المصفوفة الفارغة الأحادية البعد إلى إجراء المصفوفة البرامترية .

To call a procedure with a parameter array and supply a list of arguments
لاستدعاء إجراء مع المصفوفة البرامترية و توفير قائمة من الوسيطات

```
Call studentScores("Anne", "10", "26", "32", "15", "22", "24", "16")
```

```
Call studentScores("Mary", "High", "Low", "Average", "High")
```

- كتابة إستدعاء الإجراء بالشكل الطبيعي . مع وجوب أن تكون المصفوفة البرامترية الوسيطة الأخيرة Last Argument .
- توفير أي عدد من الوسيطات ، مفصولة بفواصل ، للمصفوفة البرامترية . يجب أن يكون نوع البيانات لكل وسيطة قابلة للتحويل ضمنيا إلى نوع عنصر (ParamArray Element Type) ParamArray .
- الفيجوال بيسك يمرر المصفوفة الأحادية البعد إلى الإجراء و الذي يحتوي على كافة القيم التي قمنا بتوفيرها .

To call a procedure with a parameter array and supply an array of arguments
لاستدعاء إجراء مع المصفوفة البرامترية وتوفير مصفوفة من الوسيطات

```
Dim JohnScores() As String = {"35", "Absent", "21", "30"}
Call studentScores("John", JohnScores)
```

- كتابة إستدعاء الإجراء بالشكل الطبيعي . مع وجوب أن تكون المصفوفة البرامترية الوسيطة الأخيرة Last Argument .
- للمصفوفة البرامترية ، نقوم بتوفير مصفوفة أحادية البعد مع نفس نوع العنصر كنوع عنصر المصفوفة البرامترية .
- الفيجوال بيسك يمرر المصفوفة إلى الإجراء .

مثال :

الأمثلة التالية توضح الإستدعاءات النموذجية إلى الإجراء studentScores المعرف في الفقرة أعلاه .

How to: Define a Procedure with an Indefinite Number of Parameters

```
Call studentScores("George")

Call studentScores("Anne", "10", "26", "32", "15", "22", "24", "16")

Call studentScores("Mary", "High", "Low", "Average", "High")

Dim JohnScores() As String = {"35", "Absent", "21", "30"}
Call studentScores("John", JohnScores)
```

- ❖ الإستدعاء الأول يهمل Omits المصفوفة البرامترية تماما و يوفر فقط الوسيطة الأولى First Argument المطلوبة . الإجراء studentScores يعالج هذا الإستدعاء كتمرير مصفوفة فارغة Empty Array .
- ❖ الاستدعاءان الثاني و الثالث يوفران قوائم الوسيطة Argument Lists بأطوال مختلفة إلى المصفوفة البرامترية . كل قائمة مثل هذه يتم تمريرها كمصفوفة من القيم .
- ❖ الإستدعاء الرابع يمرر المصفوفة إلى المصفوفة البرامترية .

أمثلة تطبيقية

التطبيق الأول

فعلى سبيل الثال للحصول على مجموع العناصر ضمن مجموعة متغيرة .

المجموعة الأولى : { 12 , 45 , 68 , 14 , 2 , 9 }
المجموعة الثانية : { 56 , 32 , 18 , 77 , 9 , 101 }

المجموعتان الأولى و الثانية لهما نفس العدد من العناصر مع الإختلاف في القيم المدرجة .

المجموعة الثالثة : { 84 , 3 , 78 , 61 , 14 , 27 }
المجموعة الرابعة : { 22 , 91 , 17 , 5 , 31 , 47, 215 , 88 , 378 , 58 }

المجموعتان الثالثة و الرابعة تختلفان في عدد العناصر مع الإختلاف في القيم المدرجة .

و الوسيطات الغير محدود العدد ParamArray تسمح لنا بتحقيق ذلك و بسلاسة تامة .

مثلا نريد إيجاد مجموع العناصر ضمن المجموعة التالية :

12 , 45 , 68 , 14 , 2 , 9

9 + 2 + 14 + 45 + 68 +12 = ?

إيصال الفكرة يرغمي على البدء من أخر خطوة ثم التراجع رويدا رويدا حتى الخطوة الأولى و بالتالي إستيعاب الشرح بصورة أفضل .
نريد تمرير مجموعة عناصر المجموعة السابقة الذكر إلى دالة أو إجراء و ليكن مثلا الإجراء التي نعطيه الإسم Sum

و أننا عند إستدعاء هذا الإجراء من أي مكان نحصل على النتيجة المرجوة :

```
Call  Sum(12 , 45 , 68 , 14 , 2 , 9)
```

ومن هنا نفهم معنى الوسيطات الغير محددة العدد ، حيث أنه بإمكاننا إضافة وسيطات أخرى حسب الحاجة إلى الدالة أو الإجراء المستخدم :

```
Call  Sum (12 , 45 , 68 , 14 , 2 , 9 , 51 , 84 , 22 , 37)
```

نقوم بفتح مشروع جديد مع إضافة زر على النموذج Form1 . وفي نافذة الكود للنموذج نقوم بتعريف إجراء بالشكل الطبيعي المألوف لحساب مجموع عناصر المجموعة مع إظهار النتيجة في الرسالة MsgBox .

تعريف الإجراء

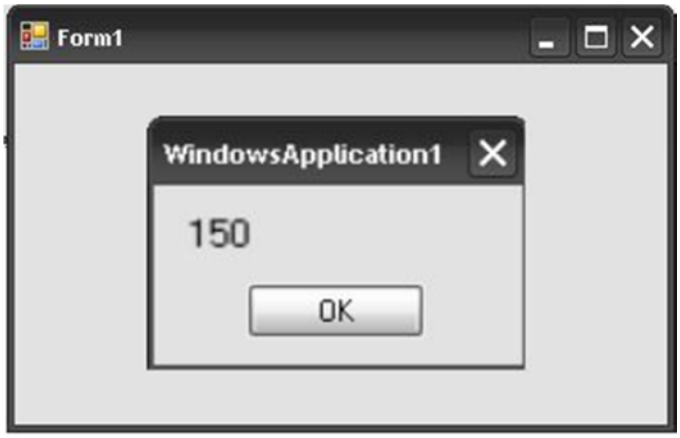
```
Sub Sum(ByVal ParamArray args() As Integer)

Dim Num As Integer

If args.Length <= 0 Then Exit Sub

For i As Integer = LBound(args) To UBound(args)

Num += args(i)
' Num = Num + args(i)

Next i

MsgBox(CStr(Num)

End Sub
```

إستدعاء الإجراء

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

' 9 + 2 + 14 + 45 + 68 +12 = 150

Call Sum(12, 45, 68, 14, 2, 9)

End Sub
```

بعدها قم بتنفيذ التطبيق و أنقر على الزر لملاحظة النتيجة :



في نفس حدث الزر قم بتغيير الإستدعاءات التالية الواحد بعد الآخر ولاحظ النتائج :

```
Call Sum(12, 45, 68, 14, 2, 9, 245, 8, 514, 42, 29)
Call Sum(12, 45, 68)
Call Sum(0)
Call Sum()
Call Sum(712, 345, 68, 34, 92, 99, 245, 8, 514, 42, 9, 8, 54, 29)
```

التطبيق الثاني

وهي دالة لإيجاد أكبر عدد ضمن مجموعة مهما كان عدد الأرقام :
إستخدام Exit Function دون إسناد قيمة للدالة فستعود بقيمة 0 إن كانت عددية Integer ، لاشى "" إن كانت حرفية String ، أو Nothing إن كانت Object .

تعريف الدالة

```
Function Max(ByVal ParamArray NumbersArray() As Integer) As Integer

    Max = 0
    Dim i As Integer

        If NumbersArray.Length <= 0 Then  Exit Function


        Max = Val(NumbersArray(0))

        For i = LBound(NumbersArray) To UBound(NumbersArray)

            If Max < Val(NumbersArray(i)) Then

                Max = Val(NumbersArray(i))

            End If

        Next i

End Function
```

إستدعاء الدالة

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

        MsgBox(Max(12, 78, 45, 255, 6, 43, 345, 68, 34))

 End Sub
```

التطبيق الثالث

وهو إجراء لإيجاد الكلمات التي تتضمن السلسلة النصية "vb" ضمن مجموعة غير محدودة من الكلمات :

تعريف الإجراء

```
Sub FindStr(ByVal ParamArray Find() As String)

Dim Word As String = "vb"

Dim Str As String

If Find.Length <= 0 Then  Exit Sub

For Each Str In Find

If Str.Contains(Word) Then

MsgBox(Str )

End If

Next

End Sub
```

إستدعاء الإجراء

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

FindStr("microsoft.vb.com", "Visual", "vbnet", "basic", "netvb2010", "Studio")

End Sub
```

التطبيق الرابع

نفس المثال السابق لكن ماذا لو أردنا في كل مرة تغيير السلسلة النصية بدلا من "vb" . سنقوم بإضافة برامتر مطلوب للإجراء عند تعريفه من نوع السلاسل النصية String .

تعريف الإجراء

```
Sub FindStr(ByVal Word As String , ByVal ParamArray Find() As String)

Dim Str As String

    If Find.Length <= 0 Then

     MsgBox("المصفوفة فارغة")

    Exit Sub

For Each Str In Find

If Str.Contains(Word) Then

MsgBox(Str )

End If

Next

End Sub
```

إستدعاء الإجراء

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

   Dim Masfofa() As String = {"microsoft.vb.com", "Visual", "comvbnet", "basic", "netvbcom2010", "Studio"}

   Dim Kalima As String = "com"

     FindStr(Kalima, Masfofa)

End Sub
```

حيث يمكننا تغيير قيم المصفوفة Masfofa() و المتغير Kalima في كل مرة عند إستدعاء الإجراء .

التطبيق الخامس

تعريف الإجراء

```
Sub ReplaceStr(ByVal Word As String, ByVal NewWord As String, ByVal ParamArray Find() As String)
    Dim Str As String
    If Find.Length <= 0 Then
        MsgBox("المصفوفة فارغة")
        Exit Sub
    End If
    For Each Str In Find
        If Str.Contains(Word) Then
            MsgBox(Str.Replace(Word, NewWord))
        End If
    Next
End Sub
```

إستدعاء الإجراء

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click

    Dim Masfofa() As String = {"microsoft.vb.com", "Visual", "comvbnet", "basic", "netvbcom2010", "Studio"}
    Dim Kalima As String = "com"
    Dim NewKalima As String = "org"

    ReplaceStr(Kalima, NewKalima, Masfofa)

End Sub
```

التطبيق السادس

إستخدام المصفوفة البرامترية مع الأدوات .
قم بإضافة مربع حوار وسرد ComboBox1 إلى نافذة الفورم رفقة الزر Button1 .

تعريف الإجراء

```
Sub ComboItems(ByVal ParamArray Args() As String) As String
    If Args.Length <= 0 Then Exit Sub
    For i As Integer = 0 To Args.GetUpperBound(0)
        ComboBox1.Items.Add(Args(i))
    Next
    ComboBox1.Text = Args(0)
End Sub
```

إستدعاء الإجراء

من حدث النقر على الزر

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim Masfofa() As String = {"microsoft.vb.com", "Visual", "comvbnet", "basic", "netvbcom2010", "Studio"}
    ComboItems(Masfofa)
End Sub
```

و لله الحمد