دروس تعليم LabVIEW

الدرس الخامس

Structures

الفهرس

هدف الدرس iii
مقدمة iv
Loops (دوال التكرار) 1
For Loop 2
كيفية ادراج For Loop 3
خاصية Auto Grow 5
حذف For Loop 6
Numeric Conversion 7
While Loop 10
كيفية ادراج While Loop 12
ملحوظات حول استخدام Terminals مع Loops 14
Shift Registers وFedback Node 17
Shift Register 18
كيفية اضافة Shift Register 18
مثال على Shift Register 20
مثال ثانى على Shift Register 22
القيم الابتدائية للـ Shift Register 24
كيفية تحديد القيم الابتدائية للـ Shift Register 24
ماذا يحدث اذا لم يتم تحديد القيم الابتدائية للـ Shift Register؟ 24
Feedback Nodes 27
ادراج Feedback Node 28
ماذا يحدث اذا لم يتم تحديد القيم الابتدائية للـ Feedback Node؟ 29
Case Structure 30
كيفية تحديد الحالات مع الانواع المختلفة للقيم الموصلة للـ Selector Terminal 33
Default Case او الحالة الافتراضية 34
اضافة وحذف حالات 38

الادخال و الاخراج من و الى Case Structure 39
Dialogs 41
Standard Dialogs-1 41
One Button Dialog 42
Two Button Dialog 43
Two Button Dialog 44
Express Dialogs-2 44
Display Message Express VI 44
Prompt User Express VI 46
The Sequence Structure 49
Flat Sequence Structure 49
Stacked Sequence Structure 49
ادراج Sequence Structure 50
Sequence Local 51
اضافة Sequence Local 52
المؤقتات Timing 55
الدوا ل الاساسية 55
1. دالة Wait (ms) 55
2. دالة Wait Until Next 56
3. دالة Tick Count (ms) 57
Express Timing Functions 58
1. Time Delay Express VI 58
2. Elapsed Time Express VI 58
Formula Node 60
كيفية ادراج Formula Node 60
اضافة متغيرات داخلة ومتغيرات خارجة 61
كتابة المعادلات داخل Forumla Node 62
Expression Node 65

هدف الدرس

- التعرف على دوال التكرار While Loop وFor Loop.
- التعرف على Shift Registers و Feedback node.
- التعرف على Case Structure.
- التعرف على Dialogs والطرق المختلفة لاخراج رسائل للمستخدم اثناء تنفيذ البرنامج.
- التعرف على Flat and Stacked Sequence Structure.
- التعرف على طرق عمل التاخيرات الزمنية وحساب الوقت فى البرامج.
- التعرف على Formula Node و Expression Node لكتابة وحساب المعادلات الرياضية داخل البرنامج.

مقدمة

Structures هى احد اهم انواع Nodes المستخدمة فى Block Diagram.

وسوف نتعرف فى هذا الدرس على Structures الاساسية فى LabVIEW :

- For Loop
- While loop
- Case Structure
- Sequences Structure
- Formula Node

Loops (دوال التكرار)

تستخدم Loops لتكرار تنفيذ جزء من الكود.ويوجد فى LabVIEW نوعان اساسيان من Loops هما For Loop و While Loop.

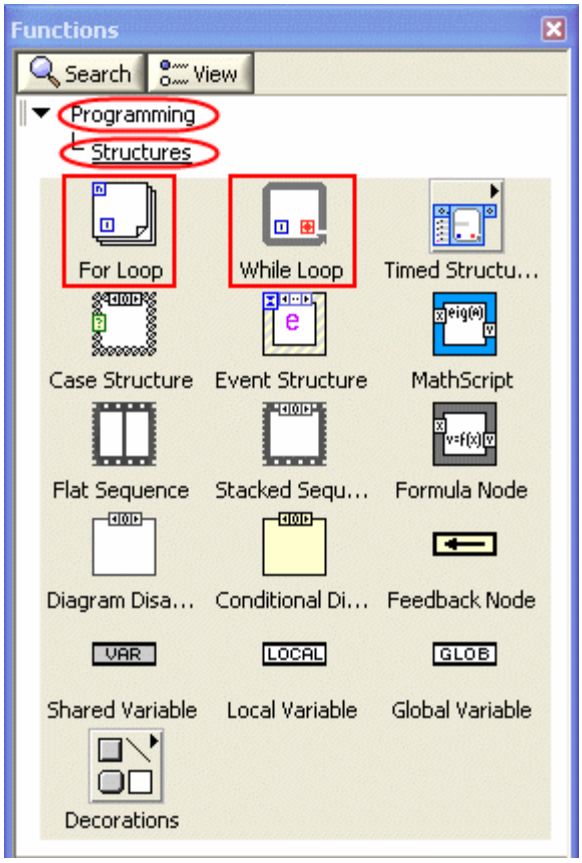
For Loop وهى لتكرار تنفيذ جزء معين من الكود عدد محدد من المرات .

مثال : تكرار جمع رقمين خمس مرات

While Loop وهى لتكرار جزء معين من الكود حتى يتحقق شرط معين اى حتى يكون هذا الشرط True او False حسب نوع الشرط المستخدم كما سنرى.

ويتم ادراج While Loop و For Loop من

Function Palette>>Programming>>Structures



For Loop

تستخدم For Loop لتكرار تنفيذ الكود الموجود بداخلها عدد محدد من المرات.

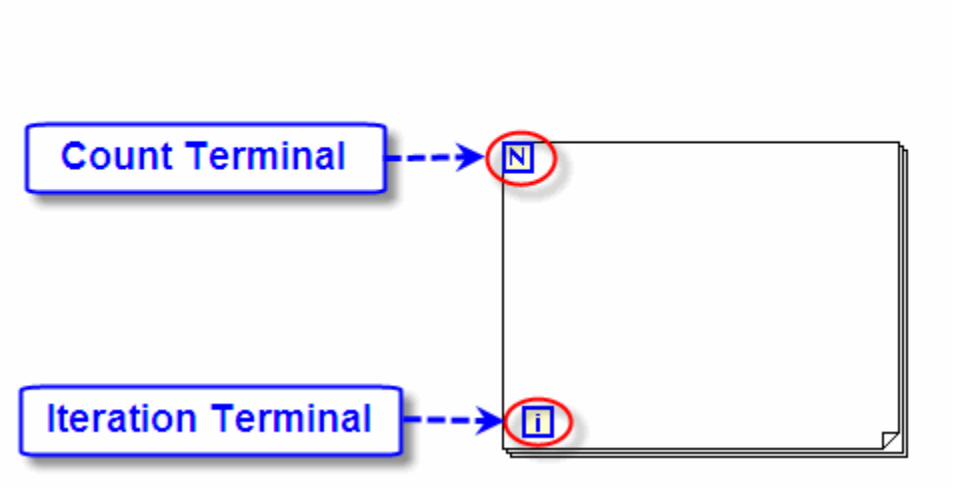
ويسمى الكود الموجود بداخل For Loop باسم SubDiagram.

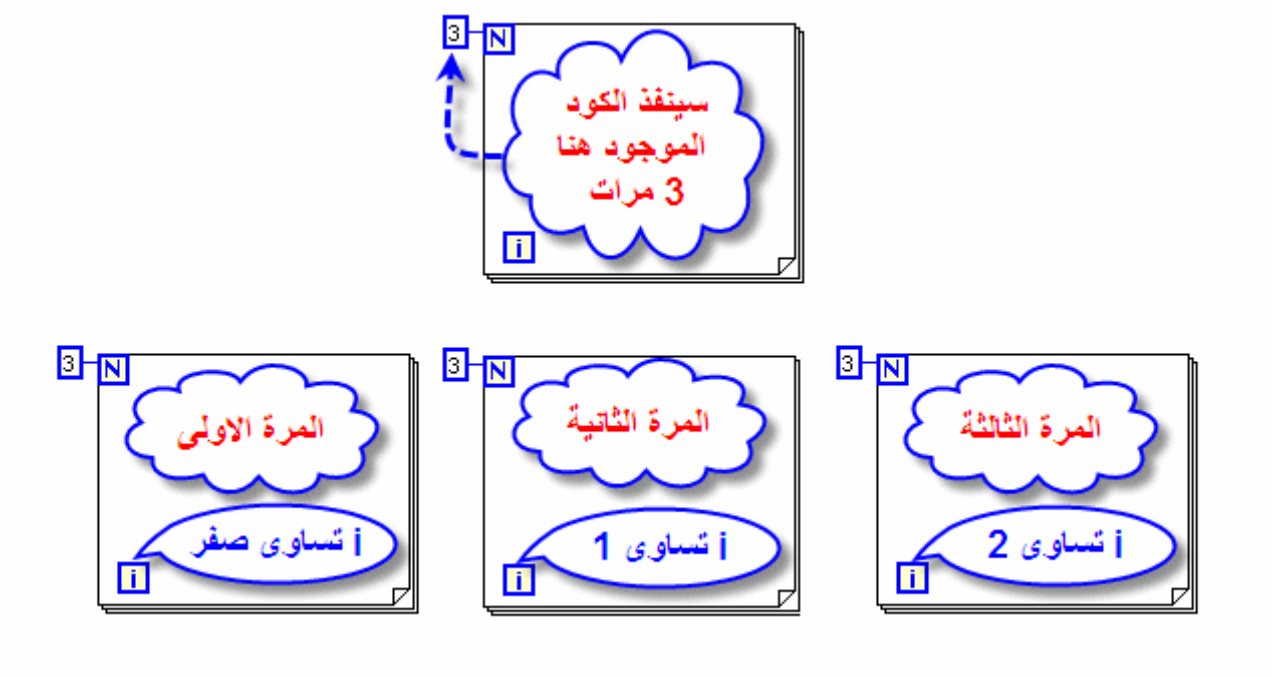
ويوجد مع For loop 2 Terminals:

N Count Terminal وهو يحدد عدد مرات تنفيذ ما بداخل For loop ويتم توصيل العدد له من الخارج واذا تم توصيل له العدد صفر فلن ينفذ ما بداخل For Loop.

i Iteration Terminal وهو يحتوى على عدد المرات التى تم تنفيذها .ففى اول مرة ينفذ فيها الكود يكون بالقيمة صفر و فى المرة الثانية يكون بالقيمة واحد وهكذا حتى يصل الى العدد N-1.

فهذا Terminal يأخذ القيم من صفر الى N-1 . حيث N يساوى العدد الموصل الى N Count Termianl (عدد المرات التى سينفذ فيها الكود) .





ملحوظة :

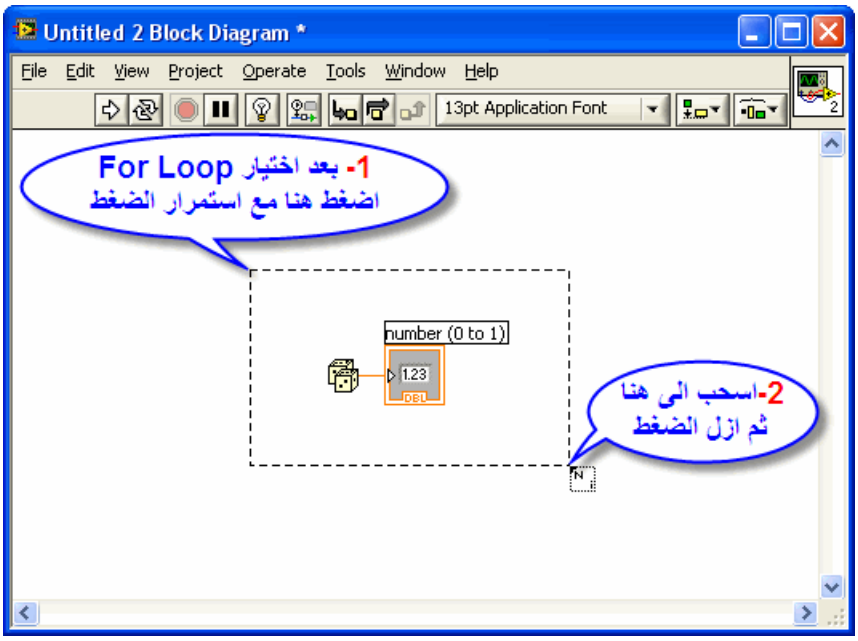
الـ Data Type لكل من N و i هو Long Integer فهما يأخذان قيم صحيحة ما بين صفر الى 2^{32}-1.لاحظ ان لونهما ازرق.

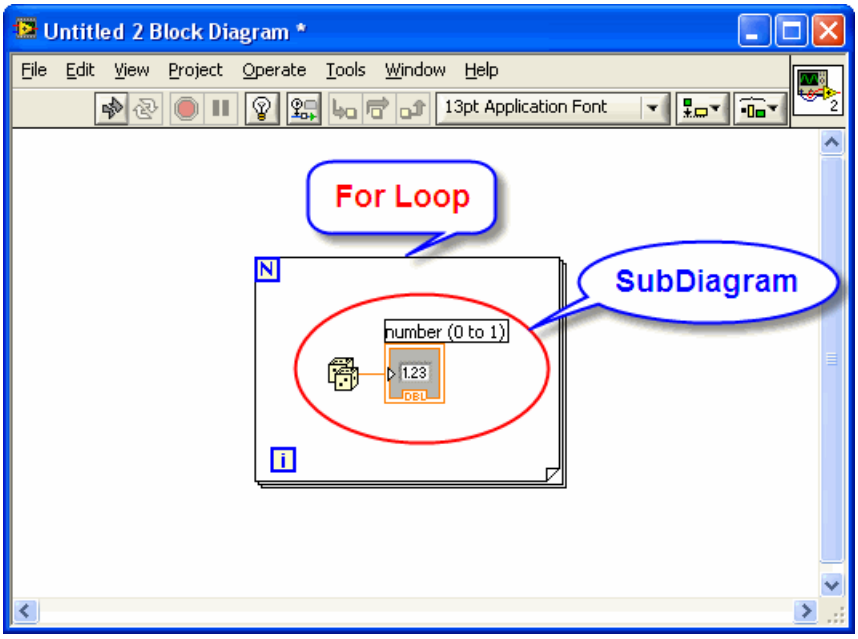
كيفية ادراج For Loop

كما ذكرنا توجد For Loop فى

Function Palettes>>Programming>>Structures

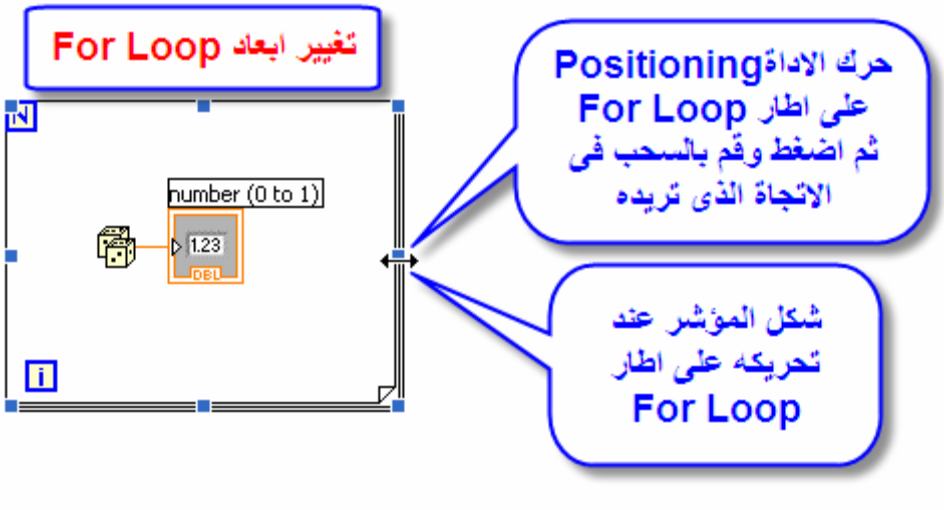
اختر For Loop ليظهر المؤشر بهذا الشكل N. ثم ابدأ فى رسم For Loop فى المكان الذى تريده.



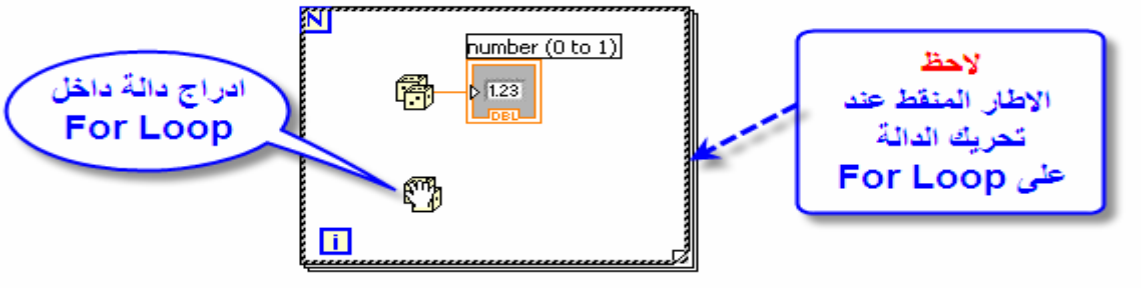


لاحظ انه :

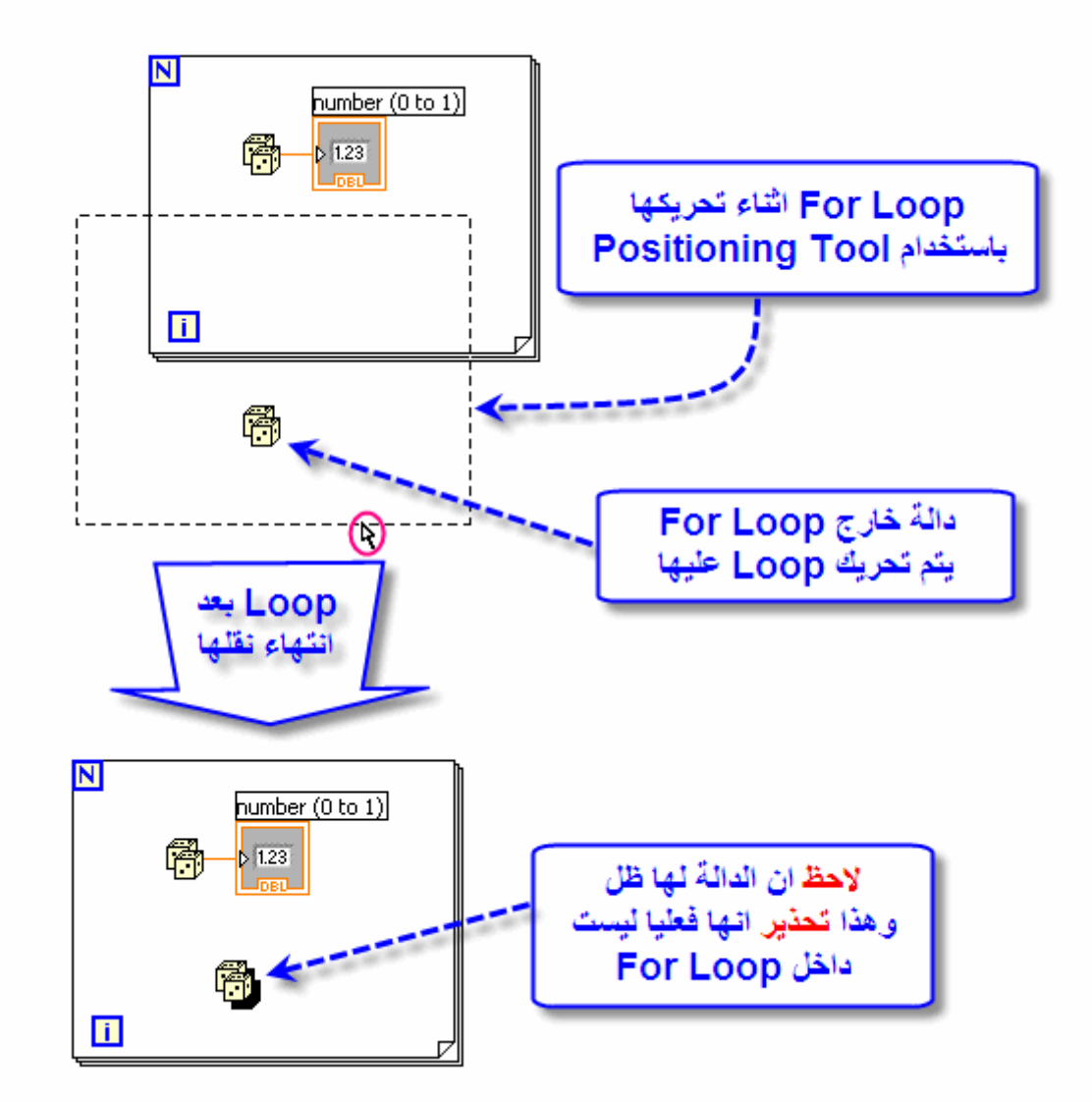
- يمكن تغيير ابعاد For Loop باستخدام الاداة Positioning وذالك بتحريكها على اطار Loop ليصبح المؤشر على شكل سهم فنضغط بالمؤشر ونسحب الاطار فى الاتجاة الذى نريده.



- يمكن ادراج اى دالة او SubVI او اى Structure داخل For Loop



- ايضا يمكن تحريك For Loop و نقلها بواسطة Positioning Tool.
- عند نقل For Loop ينتقل معا ما بداخلها.
- لاحظ انه اذا قمت بتحريك For Loop لتغطى على SubVI او دالة سيظهر ظل لهذة الدالة ليحذرك انها ليست داخل For Loop.



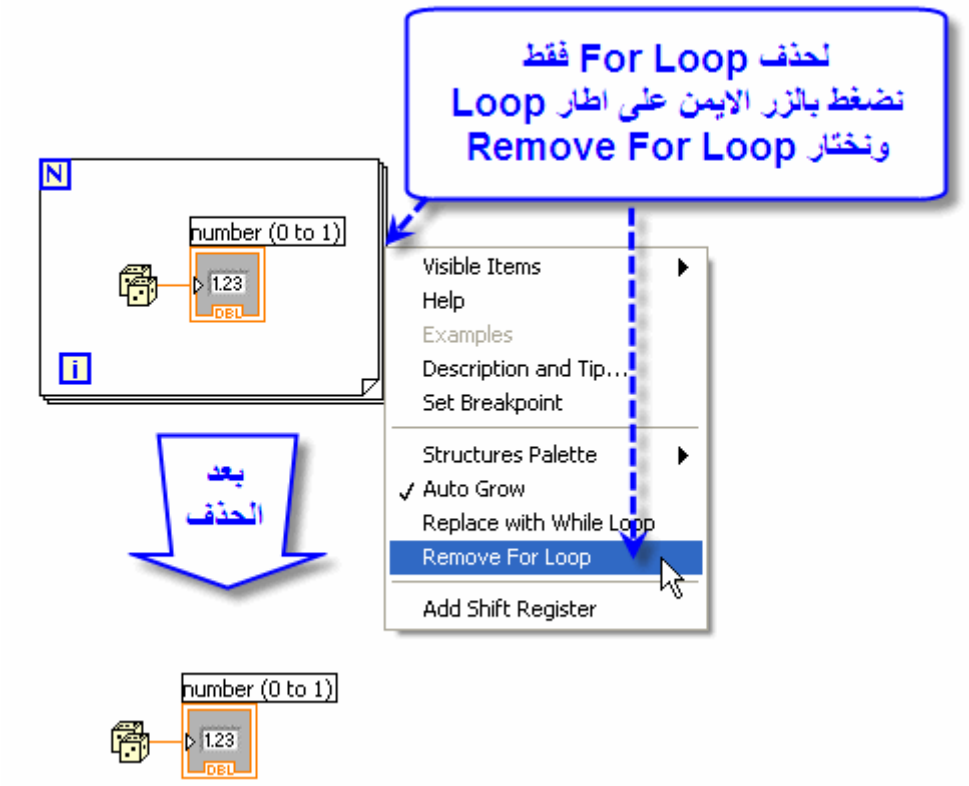
خاصية Auto Grow :

كل Structure بما فيها For Loop لها خاصية Auto Grow وهى ان For Loop تتسع تلقائيا لتسع كل ما بداخلها ويكون كله ظاهرا.

ويمكن تفعيل هذا الخاصية او الغائها وذلك بالضغط بالزر الايمن للماوس على اطار Loop واختيار او الغاء اختيار Auto Grow.

حذف For Loop :

- اذا اردت حذف For Loop بما فى داخلها اختر For Loop واضغط Delete.
- لحذف For Loop فقط والابقاء على ما بداخلها نضغط بالزر الايمن للماوس على اطار For Loop ونختار Remove For Loop

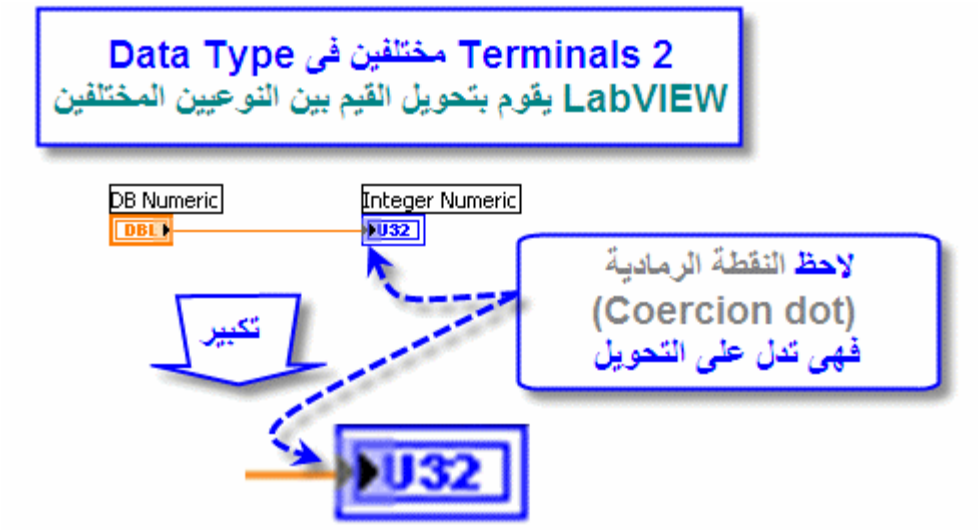


: Numeric Conversion

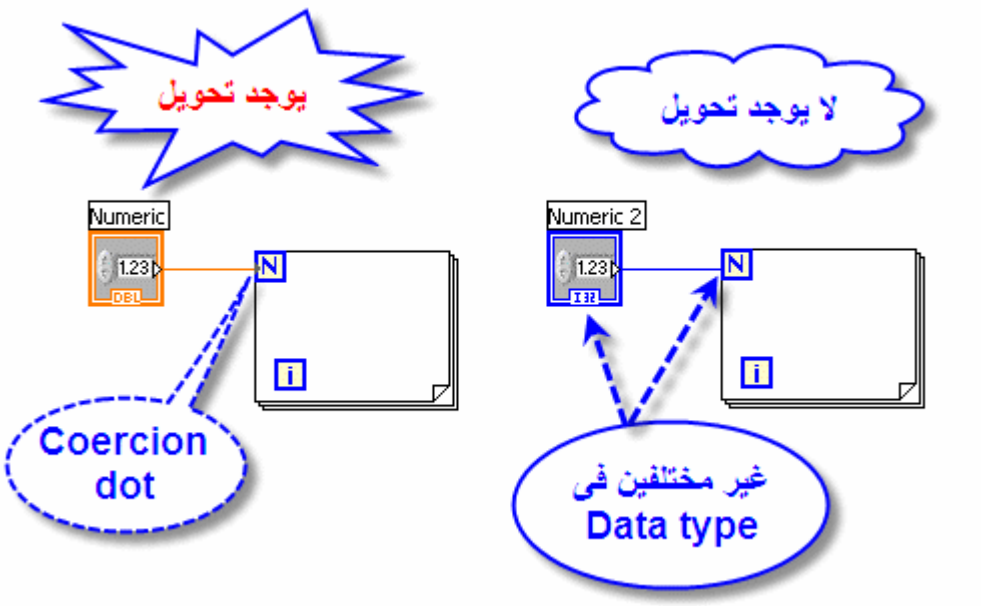
ذكرنا من قبل ان Data Type للـ Count Terminal هو Long Integer اى رقم صحيح ما بين الصفر و 2^{32}-1.

ويوجد انواع اخرى من Data Type بالنسبة للـ Control او Indicator الرقمية مثل Double-Precision وSingle-Precision .

فاذا قمت بتوصيل بين 2 Terminals مختلفيين فى نوع Data Type فسوف يقوم LabVIEW بتحويل القيم الرقمية بين النوعين المختليفين .وعندئذ يضع LabIVEW نقطة على Terminal المحول اليه و هذة النقطة تسمى Coercion dot.



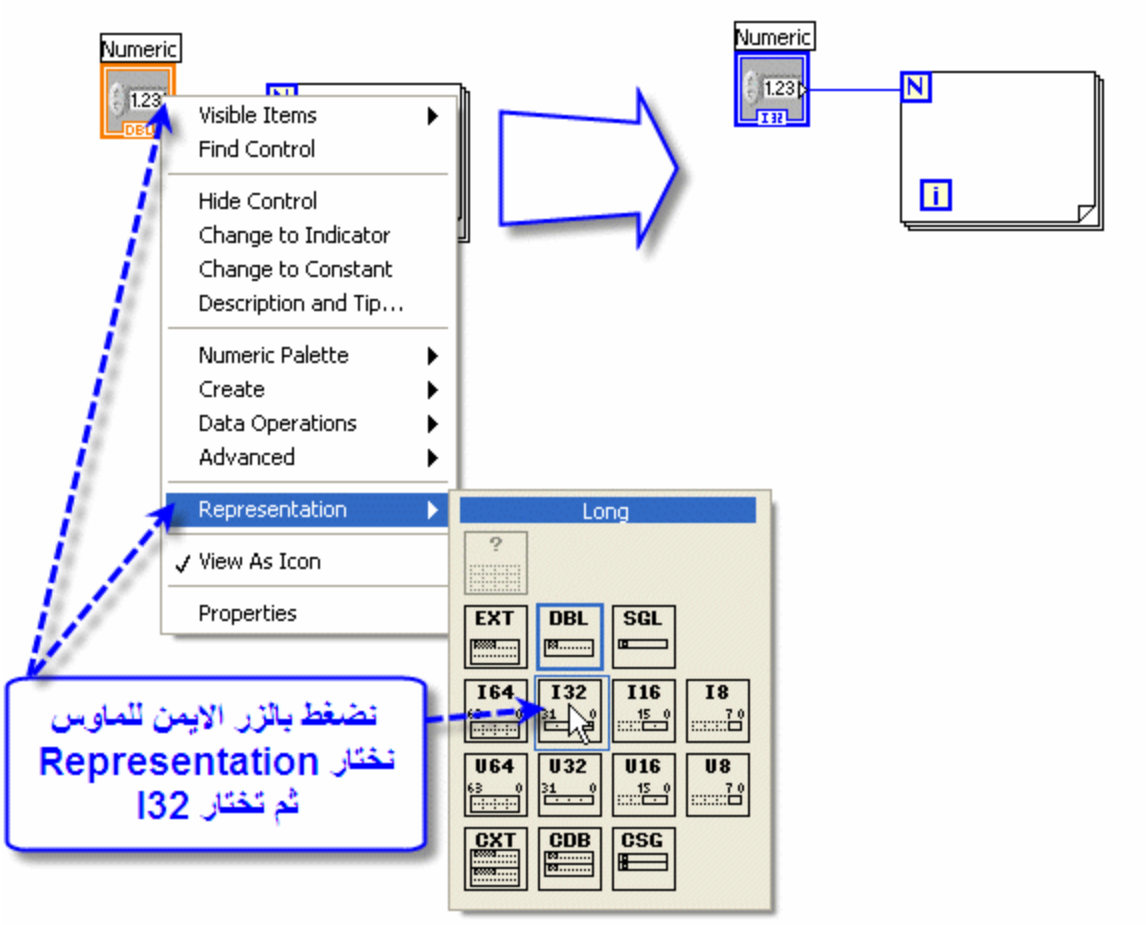
كذلك اذا قمت بتوصيل Terminal له Data type مختلف الى Count Terminal فسوف يقوم LabVIEW بعملية التحويل.



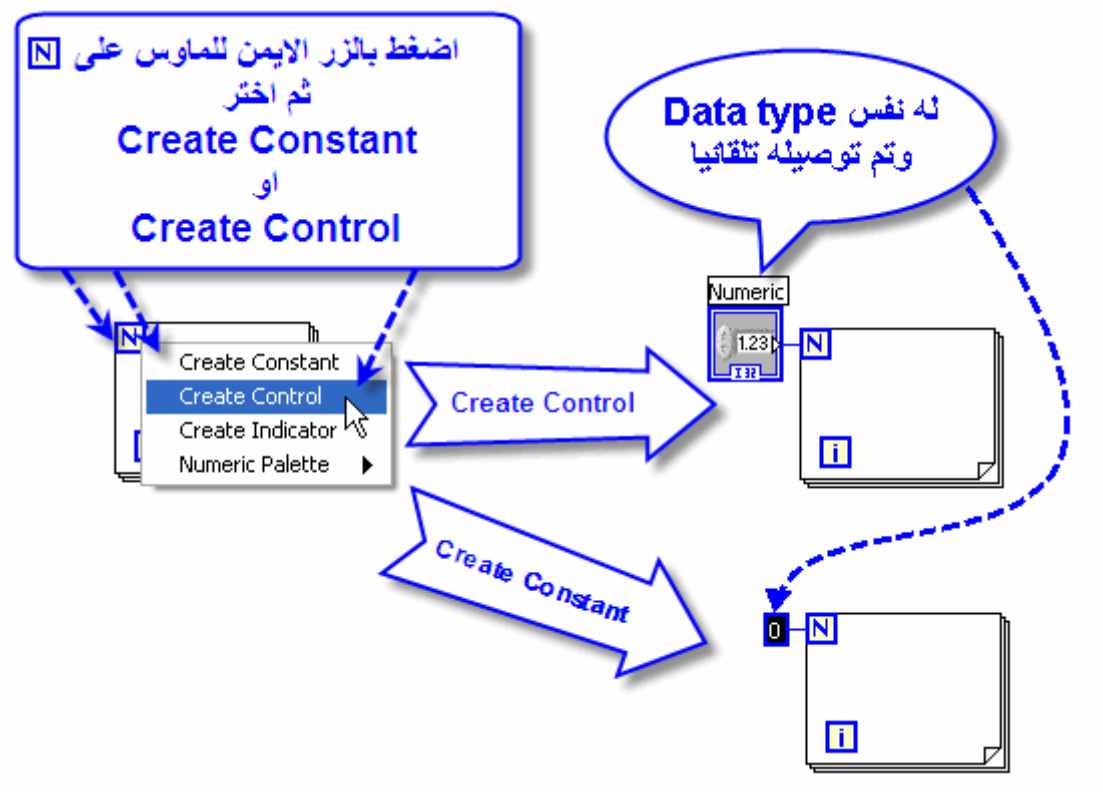
ولتجنب ذلك

1- نغير Data Type الخاص بال Terminal الموصول بـ Count Terminal

وذلك بالضغط بالزر الايمن واختيار Representation ونختار I32.

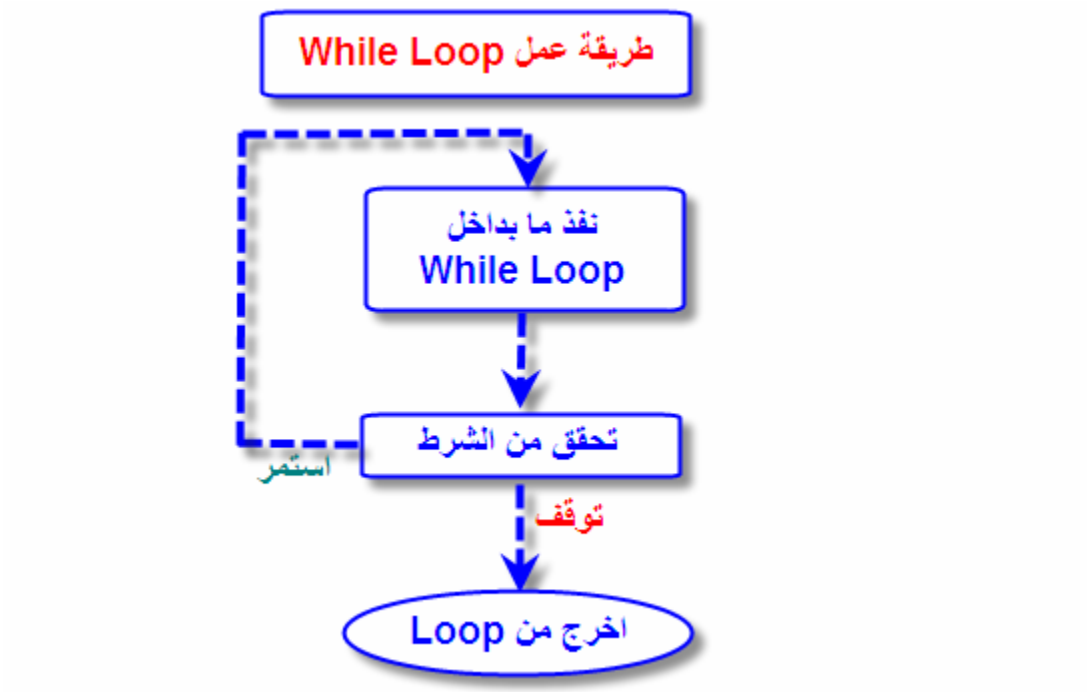


2-يمكن انشاء Constant او Control يتم توصيله تلقائيا بـ Count Terminal وله نفس Data type . وذلك بالضغط بالزر الايمن للماوس على Count Terminal واختيار Create>>Constant او Create>>Control.


اضغط بالزر الايمن للماوس على
ثم اختر
Create Constant
او
Create Control
Data type له نفس
وتم توصيله تلقائيا
Create Constant
Create Control
Create Indicator
Numeric Palette
Create Control
Create Constant
Numeric
1.23
I32
0

While Loop

هى عبارة عن Structure تعيد تكرار تنفيذ الكود الذى بداخلها حتى يتحقق شرط معين.



يوجد للـ While Loop عدد 2 Terminals وهما Iteration Terminal و Conditional Terminal

1- Iteration Terminal :

وهو مثل Iteration Terminal الموجود فى For Loop

وهو يحتوى على عدد المرات التى تم تنفيذ فيها الكود داخل While Loop.ففى اول مرة ينفذ فيها الكود يكون بالقيمة صفر وفى كل مرة يعاد فيها تنفيذ الكود يزداد واحد.

2-Conditional Terminal:

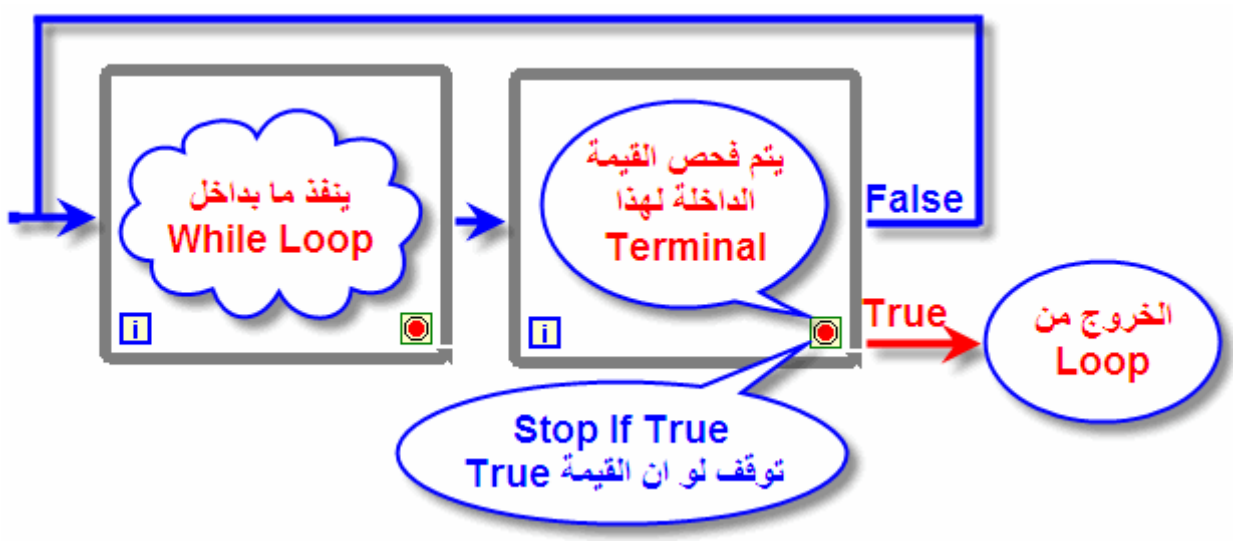
وهذا Terminal يأخذ قيم Boolean (True او False) .ويتم اعادة تنفيذ ما بداخل While Loop حتى يكون الداخل الى هذا Terminal اما القيمة True او False وذلك يتوقف على نوع Conditional Terminal.

ويوجد نوعان لهذا Terminal:

أ- Stop If True:

يتوقف تكرار تنفيذ ما بداخل While Loop اذا كانت القيمة الداخلة لهذا Terminal هى True.

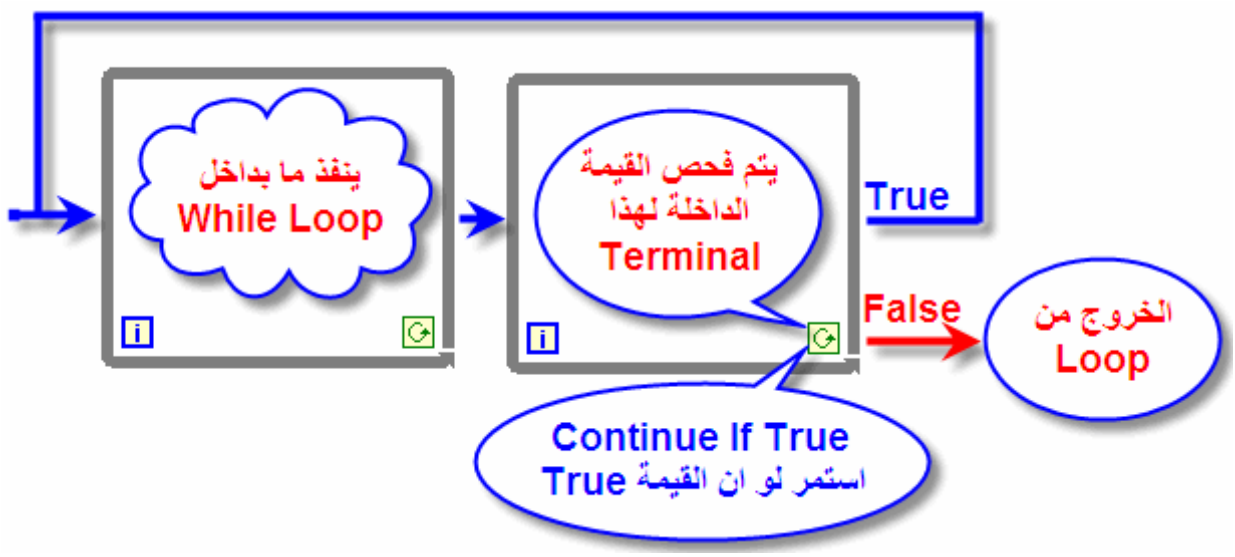
ففى كل مرة ينفذ ما بداخل While Loop ثم يتم فحص القيمة الداخلة لهذا Terminal فاذا كانت True يتم الخروج من While Loop واذا كانت False يعاد تنفيذ ما بداخل While Loop.



2- Continue if True :

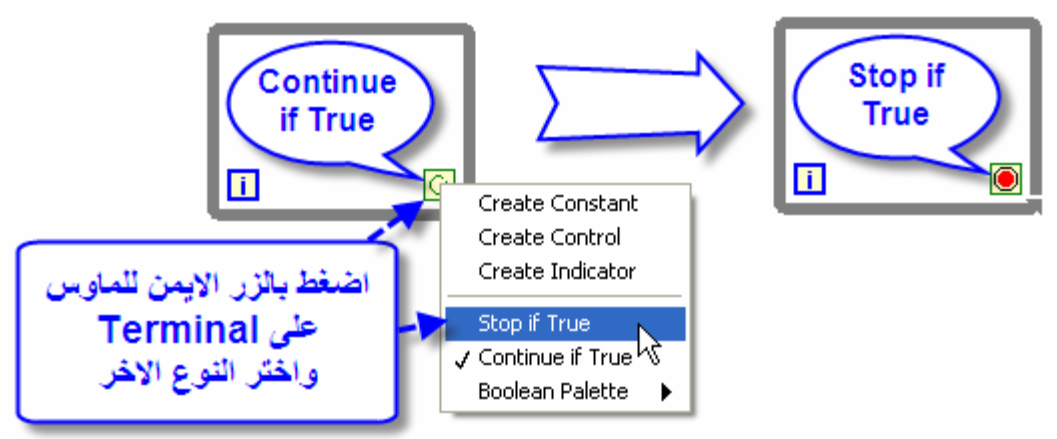
يعاد تنفيذ ما بداخل While Loop اذا كانت القيمة الداخلة لهذا Terminal هى True.

ففى كل مرة ينفذ ما بداخل While Loop ثم يتم فحص القيمة الداخلة لهذا Terminal فاذا كانت True يعاد تنفيذ ما بداخل While Loop واذا كانت القيمة الداخلة False يتم الخروج من While Loop.



لاحظ انه:

- فى اى حالة ينفذ ما بداخل While Loop مرة واحدة على الاقل.
- يمكن التحويل بين 2 Terminals وذلك بالضغط بالزر الايمن على Terminal واختيار النوع الاخر.



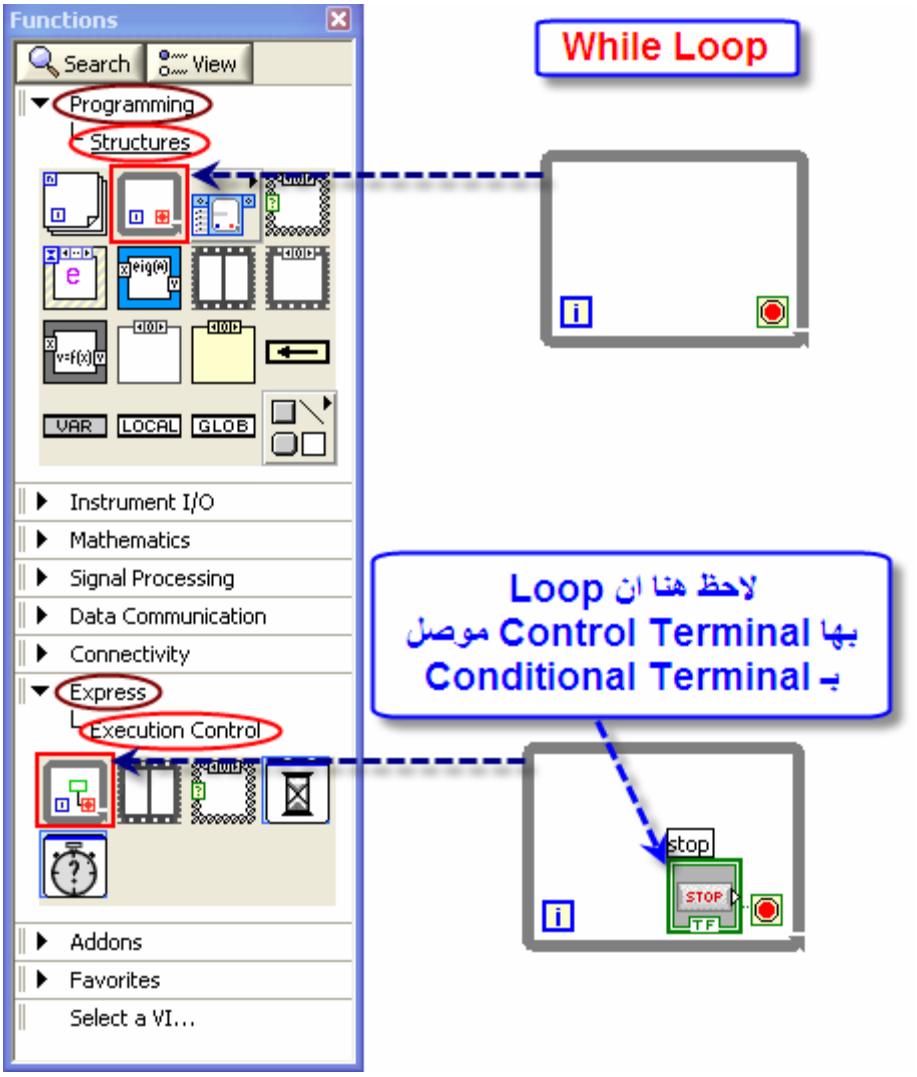
كيفية ادراج While Loop :

يمكن ادراج While Loop من

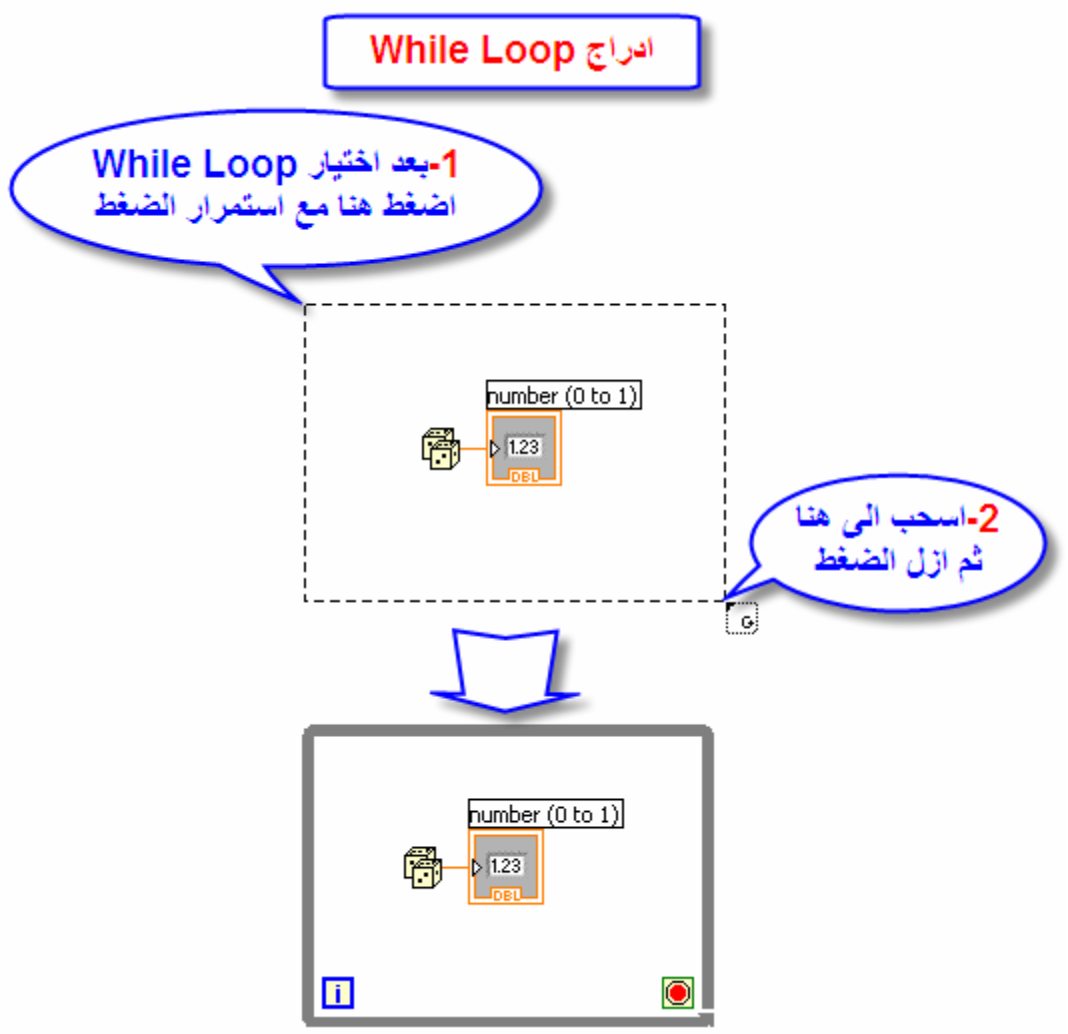
Function Palettes>>Programming>>Structures

او من

Function Palettes>>Express>>Execution Control



ومثل For Loop لادراج While Loop نختارها من Function Palette ليظهر المؤشر بهذ الشكل ثم نقوم برسم While Loop.



تتفق While Loop مع For Loop فى :

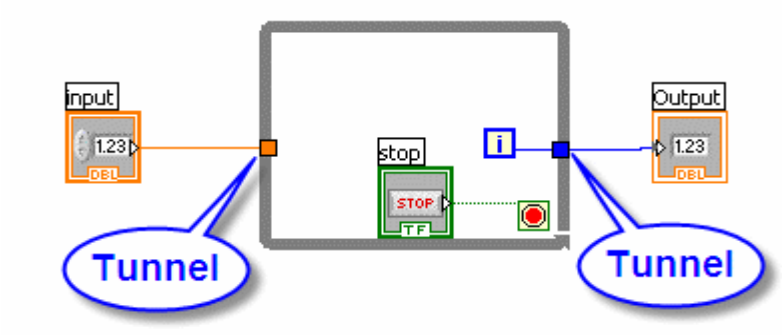
1-كيفية تحريك Loop ونقلها وتغيير ابعادها

2- طريقة حذف Loop

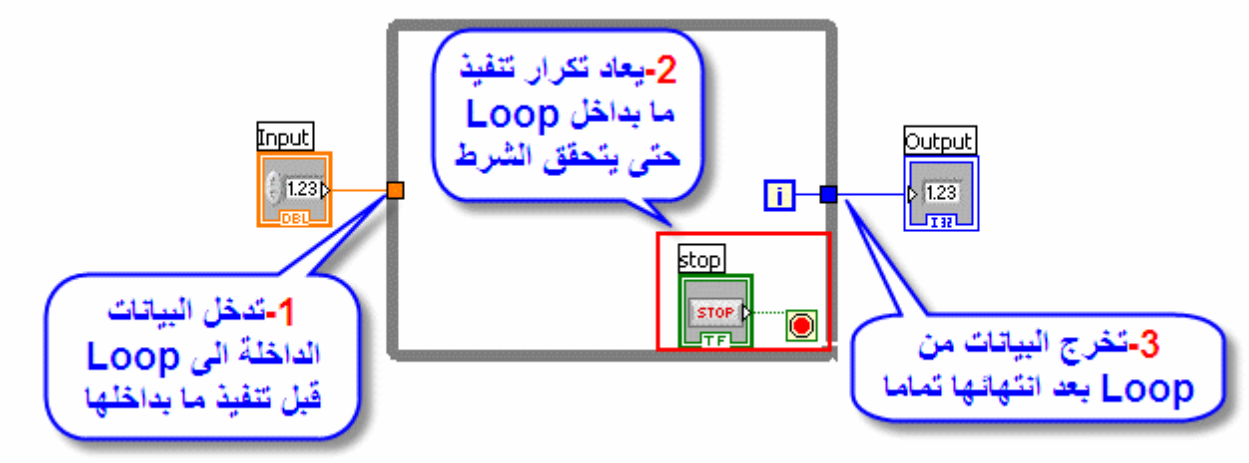
3- خاصية Auto Grow

ملحوظات حول استخدام Terminals مع Loops :

تدخل البيانات و تخرج من والى Loops من خلال مربع صغير يسمى Tunnel



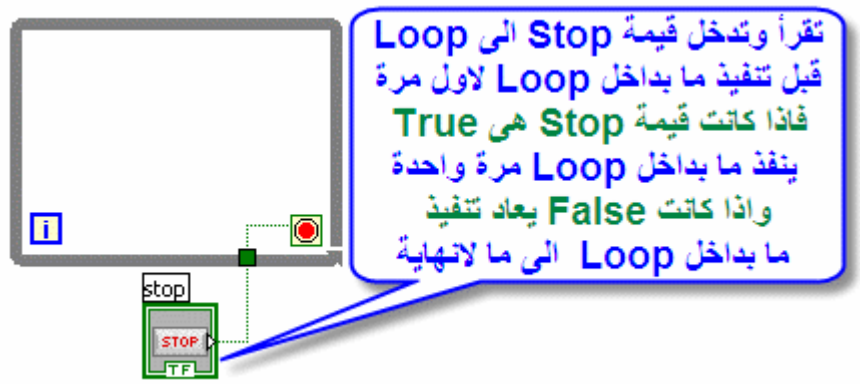
وطبقا لمفهوم Dataflow فإن البيانات الداخلة للـ Loop تدخل الى Loop قبل تنفيذ ما بداخل Loop اول مرة.وتخرج البيانات من Loop بعد تنفيذ ما بداخل Loop اخر مرة اى بعد انتهاء تكرار تنفيذ ما بداخل Loop.



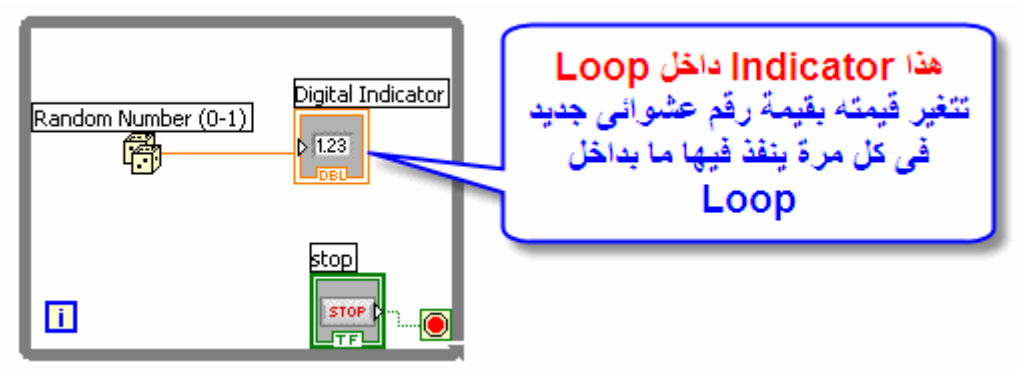
بعض الامثلة :

- Terminalداخل Loop وموصل بـ Conditional Terminal

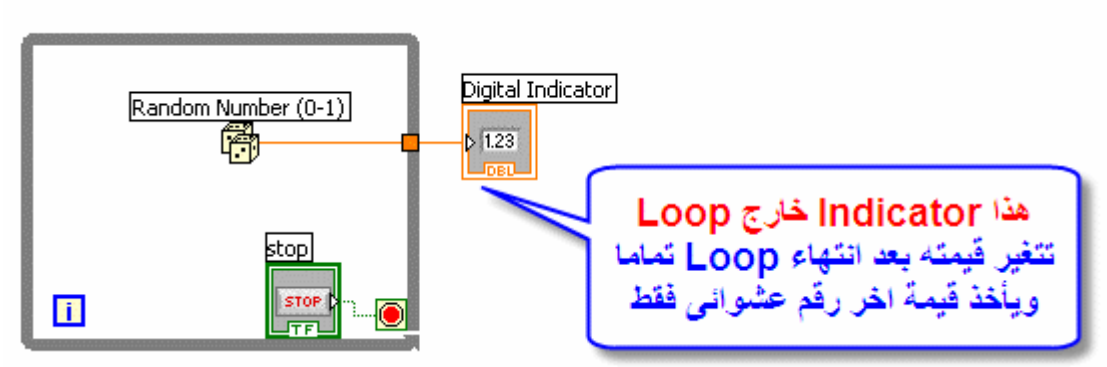
- Terminalخارج Loop وموصل بـ Conditional Terminal



- Indicator Terminal داخل Loop

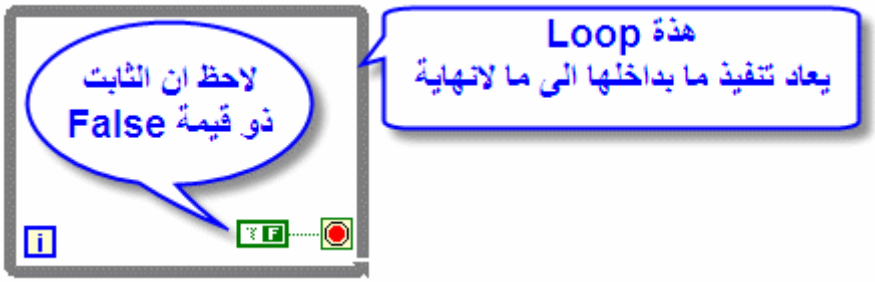


- Indicator Terminal خارج Loop وموصل بـ Loop

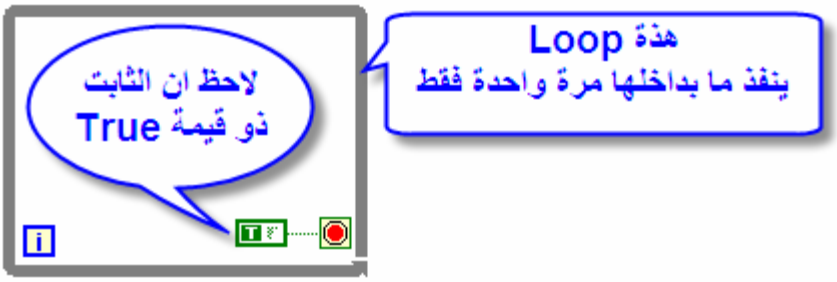


امثلة اخرى:

- توصيل ثابت قيمته False الى Conditional Terminal:Stop If True



- توصيل ثابت قيمته True الى Conditional Terminal:Stop If True



Fedback Nodeو Shift Registers

عندما تتعامل Loops سواء For Loop او While Loop ستجد انك فى حاجة لنقل البيانات من المرات السابقة التى تم تنفيذ فيها ما بداخل Loop الى المرة التى ينفذ فيها ما بداخل Loop الان.

وللتوضيح سوف نسمى تنفيذ ما باخل Loop باسم تنفيذة وسوف نعطيها رقم يدل على ترتيبها فى التنفيذ فمثلا اول مرة ينفذ فيها ما بداخل Loop نطلق عليها تنفيذة1 وثانى مرة نطلق عليها تنفيذة2 وهكذا تنفيذ3 وتنفيذة4 ... الخ.

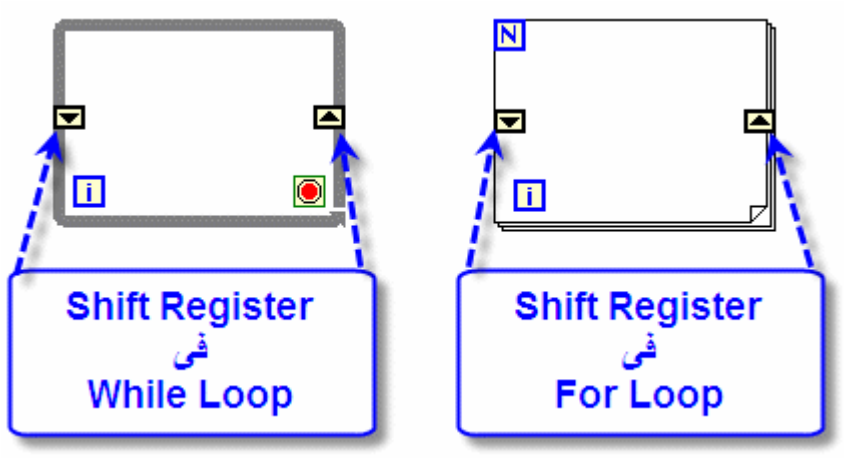
ففى كثير من الاحيان نحتاج الى نقل البيانات من تنفيذة الى التنفيذة التى تليها مثل نقل بيانات من تنفيذة1 الى تنفيذة2 وهكذا.

وهناك طريقتين للحصول على بيانات من التنفيذات السابقة هما Shift Register و Feedback Node.

Shift Register

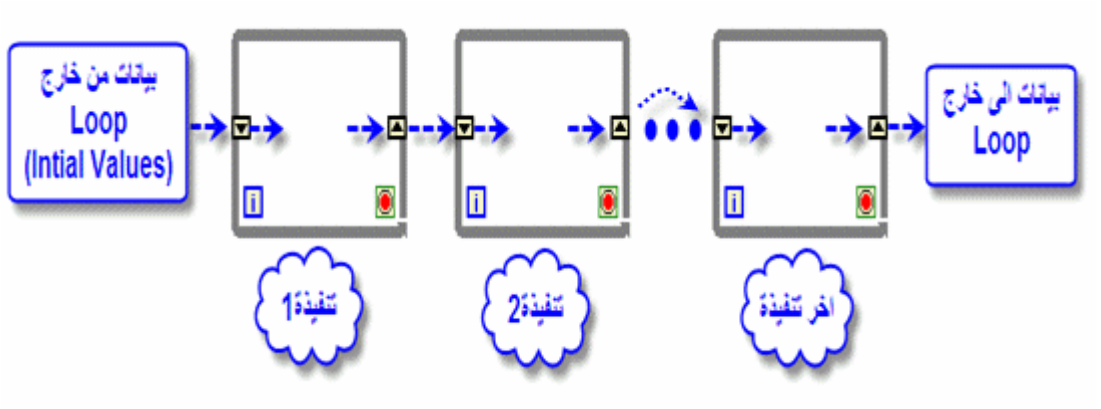
تنقل Shift Register البيانات من تنفيذة للـ For loop او While Loop الى التنفيذة التى تليها.

ويتكون Shift Register من هذين الشكلين ▼ و▲ .



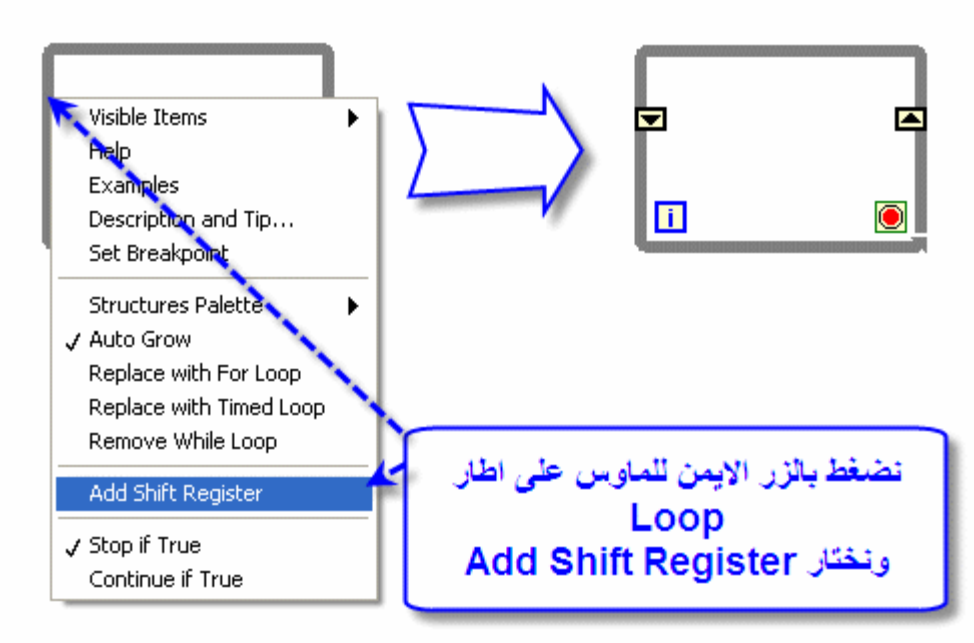
تنقل ▼ البيانات من التنفيذة السابقة الى التنفيذة الحالية.او تنقل بيانات من خارج Loop الى التنفيذة الاولى.

وتنقل ▲ البيانات من التنفيذة الحالية الى التنفيذة التالية.او تنقل البيانات من اخر تنفيذة الى خارج Loop.



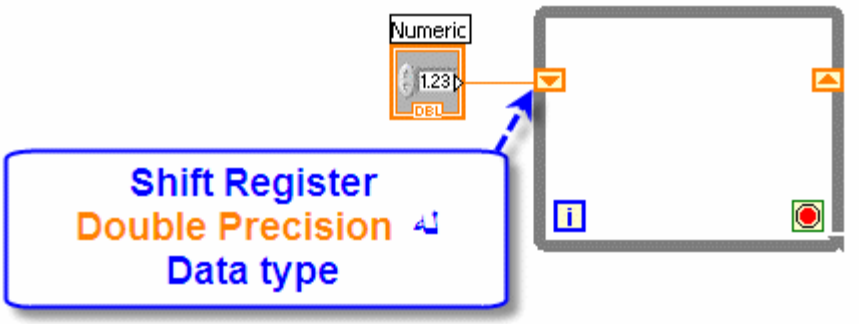
كيفية اضافة Shift Register :

تضاف Shift Register بالضغط بالزر الايمن للماوس على اطار Loop واختيار Add Shift Register.



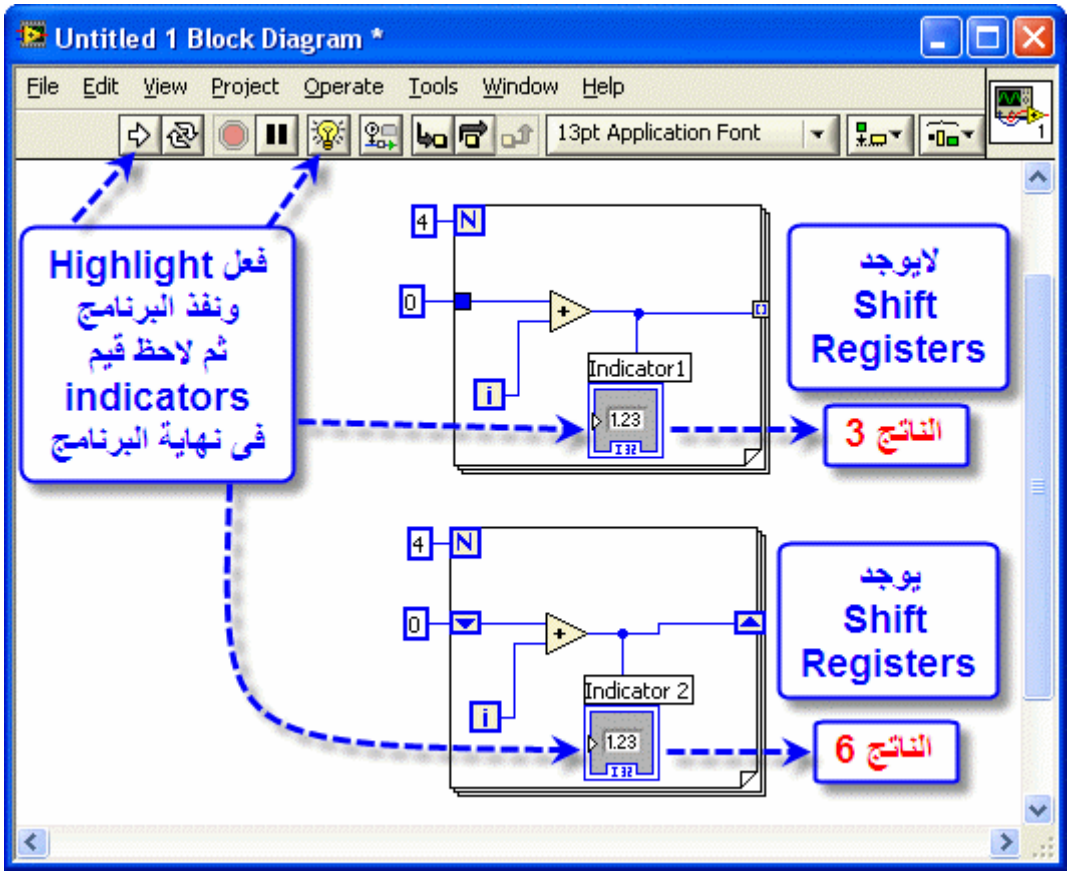
يمكن اضافة اكثر من Shift Register بنفس الطريقة.

يمكن ان ينقل Shift Register اى نوع من البيانات (اى Data type) رقم او Boolean او نص او مصفوفة ولكن يجب ان تكون نوع البيانات الموصله للـ ⊟ و ⊞ واحدة.



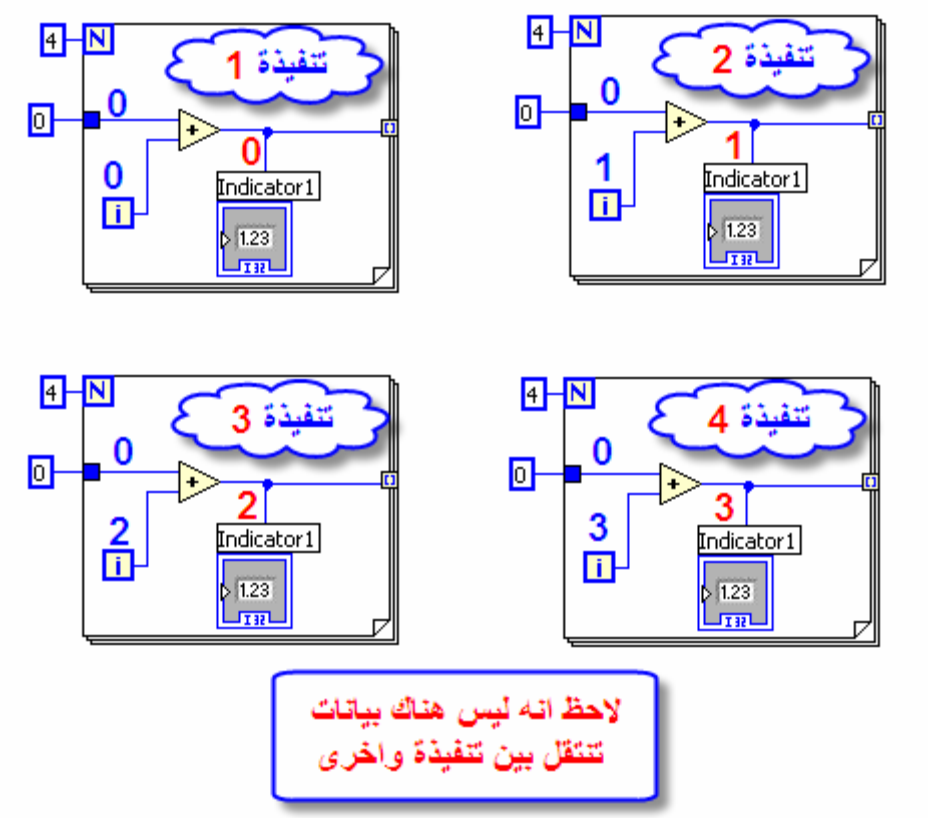
مثال على Shift Register :

نفذ البرنامج التالى باستخدام Execution Highlighting

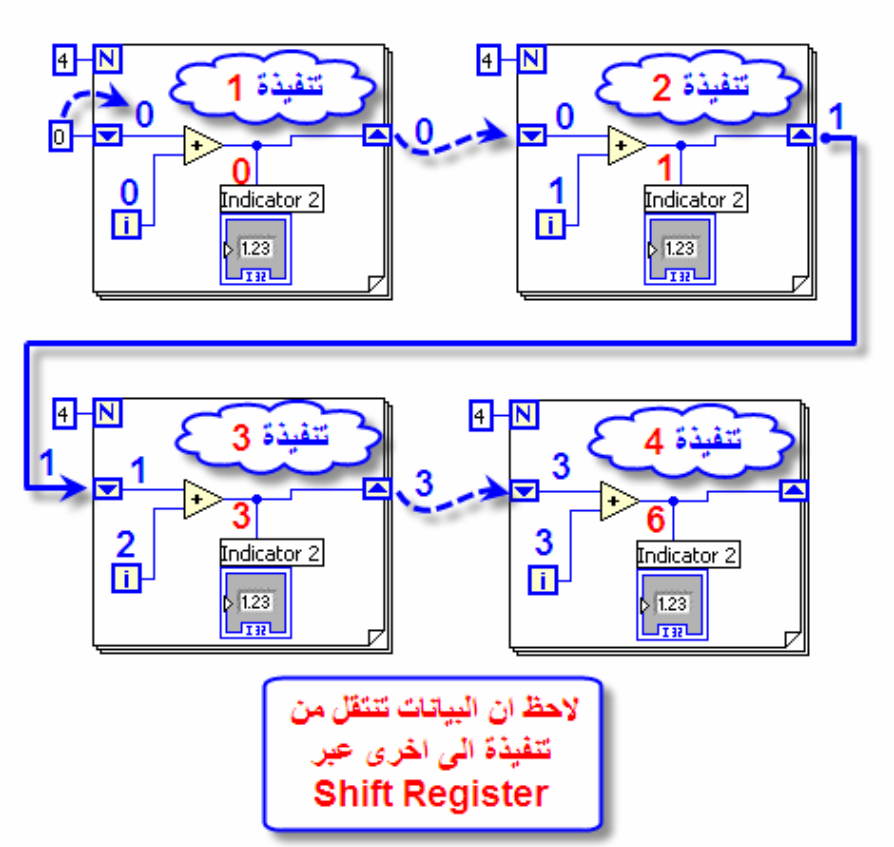


تلاحظ ان كلتا For Loops يعاد تنفيذ ما بداخلها اربع مرات.

وان خرج Indicator1 فى For Loop الاولى هو 3وذلك لاننا لم نستخدم Shift Registers فلم تنتقل النتائج بين التنفيذات الاربعة وكان تنفيذ كل واحدة منها مستقل.



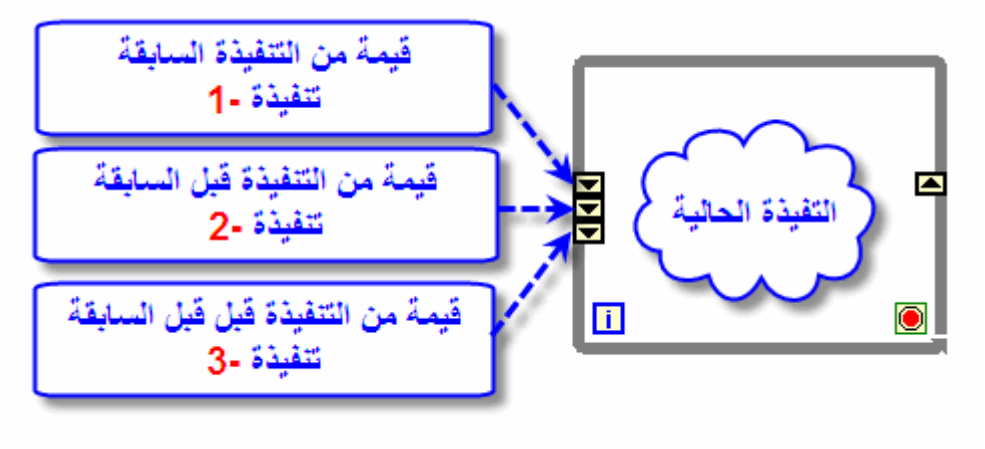
اما خرج Inidcator2 فى For Loop الثانية هو 6 وذلك لاننا استخدمنا Shift Register التى تنقل نتيجة الجمع بين التنفيذة والتى تليها.



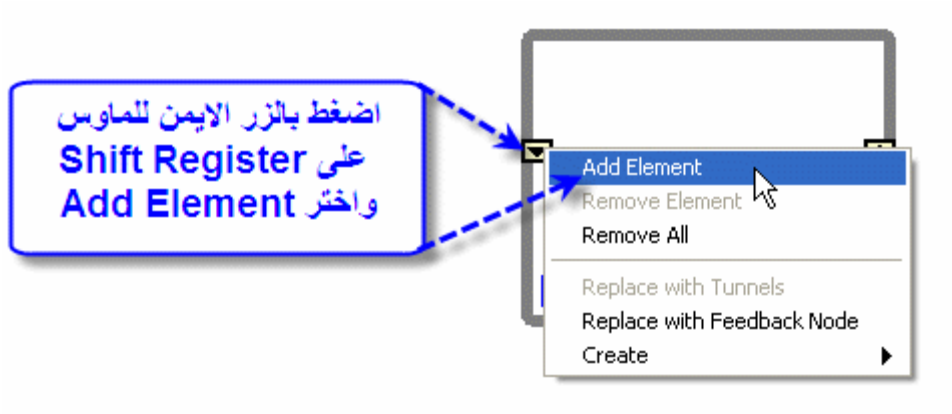
علمنا ان Shift Register تحتفظ بالبيانات من التنفيذة السابقة .

يمكن ايضا ان تحتفظ Shift Register بالبيانات من عدة تنفيذات سابقة .

وذلك يتم باضافة Shift Register Terminals على الحافة اليسرى للـ Loop

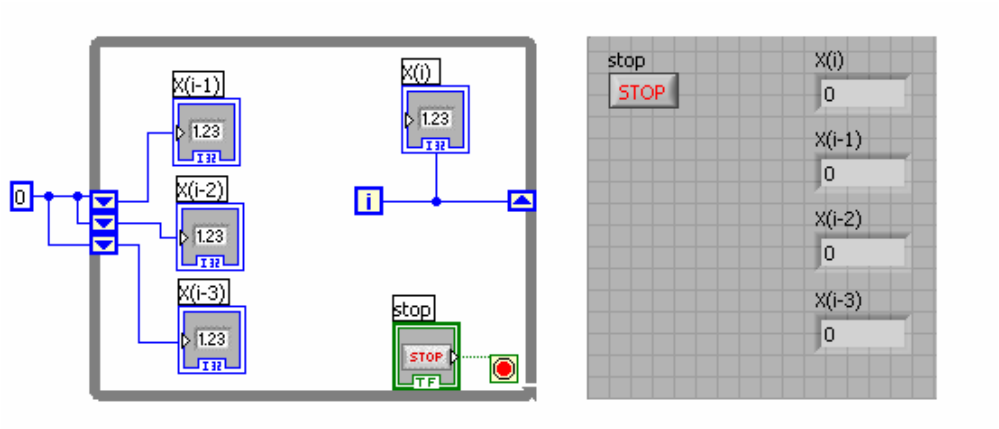


ويتم اضافة Shift Register Terminals بالضغط بالزر الايمن على Shift Register واختيار Add Element ولاضافة Terminal اخر نكرر هذة العملية.

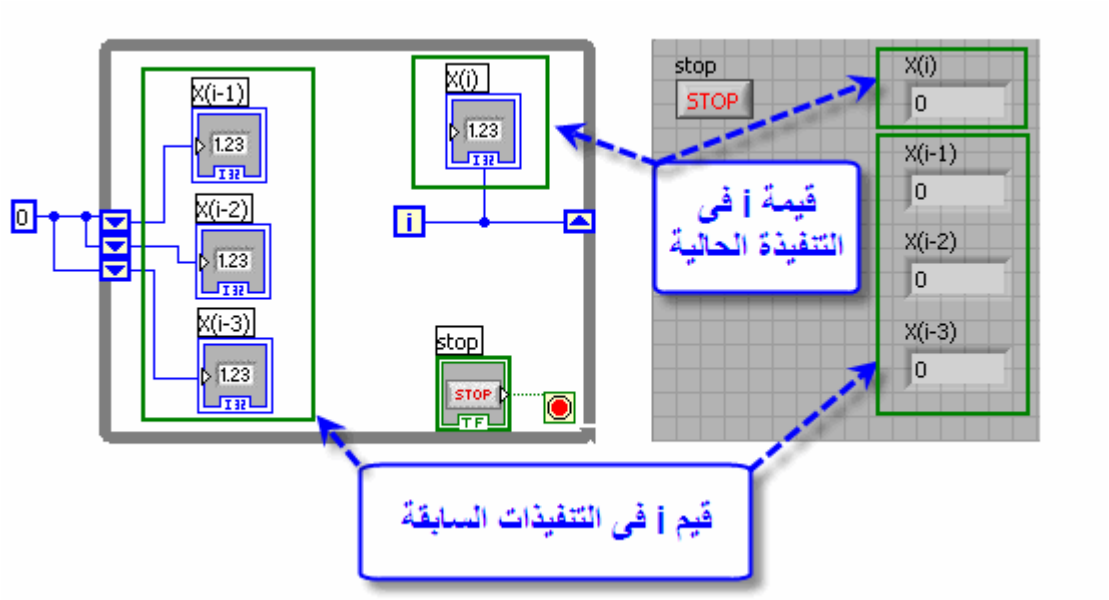


مثال ثانى على Shift Register:

كون المثال التالى والذى يتكون من While Loop بها Shift Register و مفتاح (Stop Control) وعدد اربعة Indicator وهم X(i) و X(i-1) و X(i-2) و X(i-3)



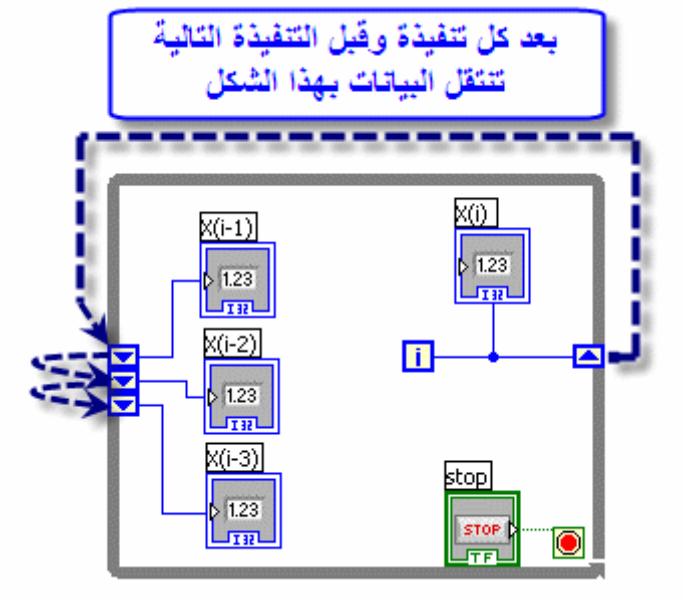
وفى هذا المثال يعرض X(i) قيمة i فى التنفيذة الحالية وتعرض X(i-1) وX(i-2) وX(i-3) قيم i من التنفيذات السابقة.



باستخدام Highlight Execution قم بتنفيذ البرنامج

1-تجد القيم المبدئية لـ X(i) وX(i-1) وX(i-2) وX(i-3) هى صفر.

2-عند انتهاء كل تنفيذة وقبل التنفيذة التالية تنتقل قيمة X(i-2) الى X(i-3) وتنتقل قيمة X(i-1) الى X(i-2) وايضا تنتقل القيمة الحالية X(i) الى X(i-1) كما بالشكل.



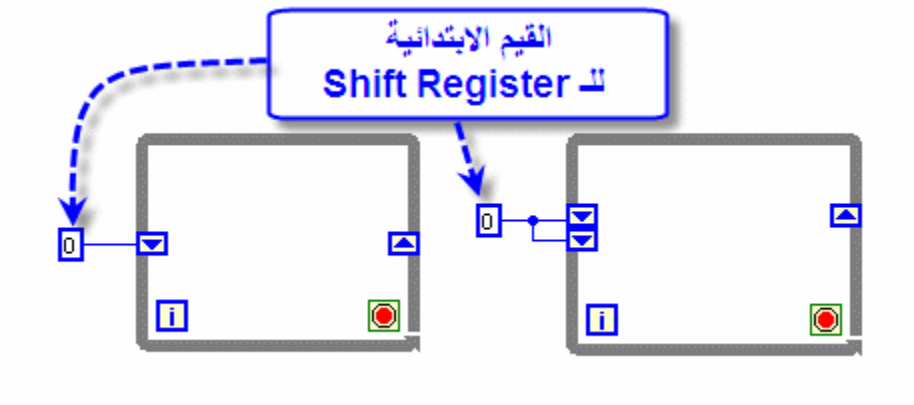
وهذة هى النتائج للتنفيذات السته الاولى

القيم الابتدائية للـ Shift Register :

هى قيم Shift Register قبل الدخول فى تنفيذ Loop.

كيفية تحديد القيم الابتدائية للـ Shift Register (Initialize Shift Register) :

تستطيع تحديد القيم الابتدائية Shift Register برط Constant او Control بـ Terminals الموجود على الطرف الايسر.

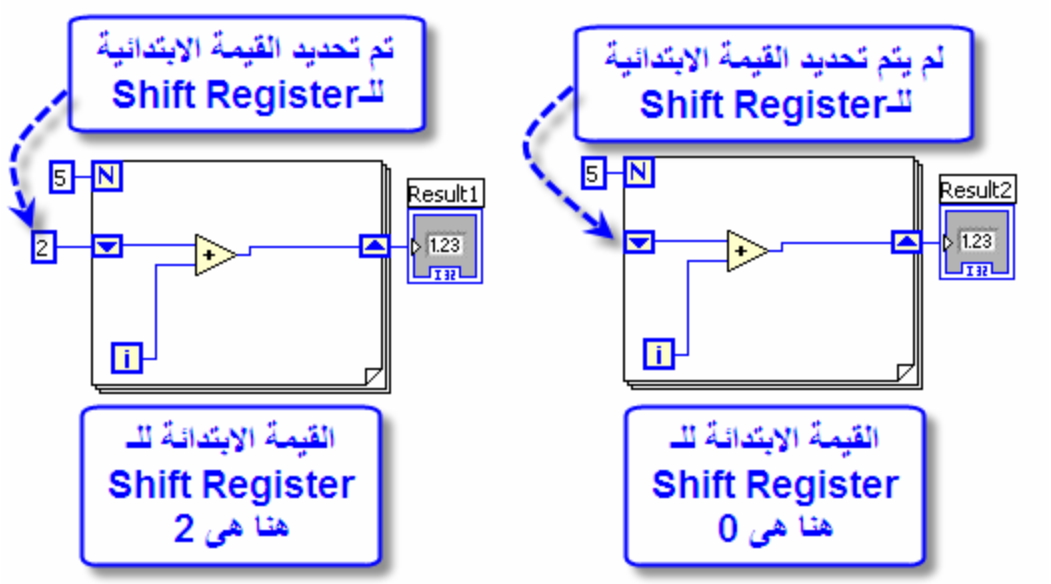


ماذا يحدث اذا لم يتم تحديد القيم الابتدائية للـ Shift Register؟

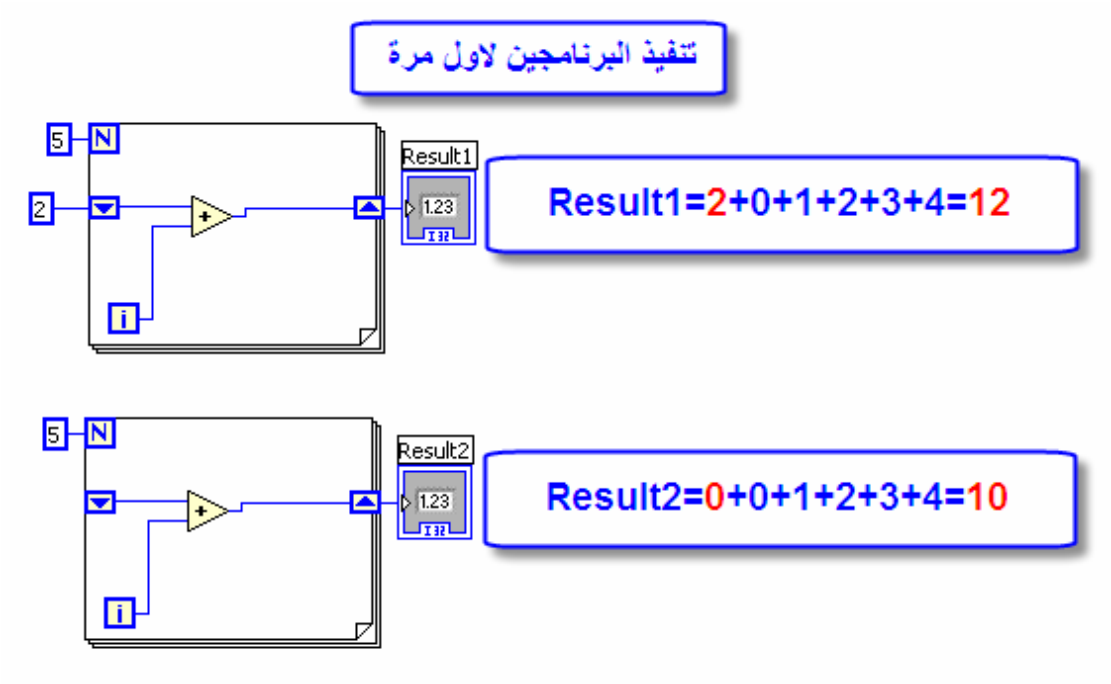
سوف تأخذ Shift Register القيم الافتراضية الخاصة بنوع البيانات التى تنقلها Shift Register اى مثلا اذا كانت تلك البيانات رقمية(Numeric) فسوف تكون القيم الابتدائية هى صفر واذا كانت البيانات ارقام ثنائية (Boolean) فسوف تكون القيم الابتدائية هى False.

و يأخذ Shift Register هذة القيم الابتدائية قبل تنفيذ البرنامج لاول مرة ولكن بعد انتهاء البرنامج تماما يحتفظ Shift Register باخر قيمه له لتكون القيم الابتدائية عند تنفيذ البرنامج للمرة الثانية.

والمثال التالى يوضح الفرق بين تحديد القيم الابتدائة وعدم تحديدها.

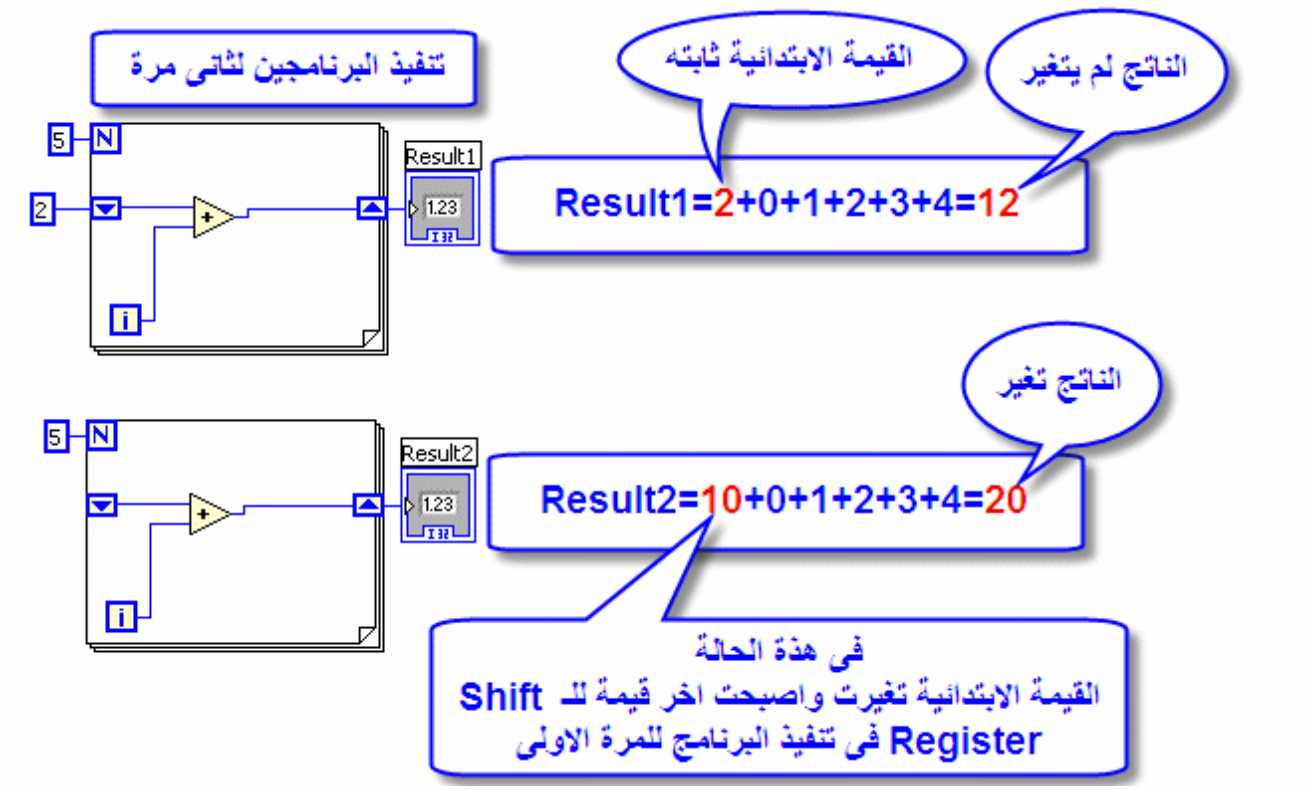


عند تنفيذ البرنامجين لاول مرة سوف تكون القيم الموجودة فى Result1و Result2 كالتالى



ولكن فى اعادة تنفيذ البرنامج نلاحظ انه فى الحالة الاولى القيمة الابتدائية ثابتة ولم تتغير

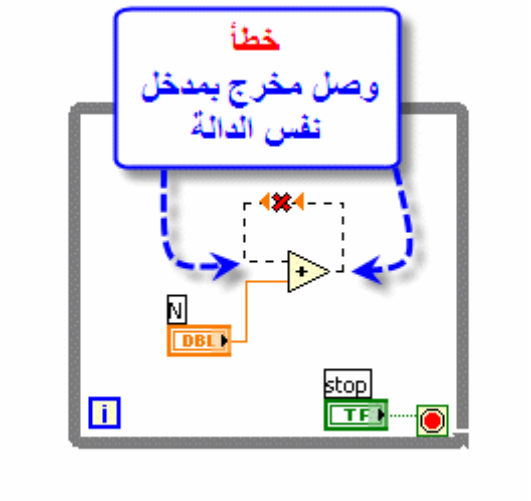
اما فى الحالة الثانية نجد ان Shift Register احتفظ باخر قيمة له من تنفيذ البرنامج لاول مرة وهى القيمة 10 و اخذ هذة القيمة كقيمة ابتدائية فى تنفيذ البرنامج للمرة الثانية لذلك الناتج تغير من 10 فى المرة الاولى الى 20 فى المرة الثانية.



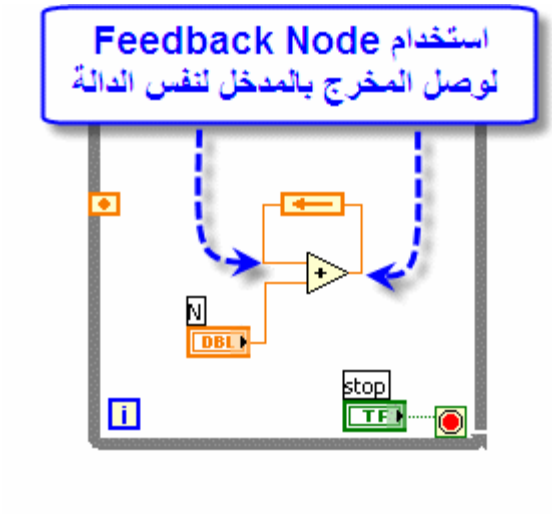
Feedback Nodes

تستخدم Feedback Node فى While Loop او For Loop عندما نريد ربط بين مخرج SubVI او دالة او مجموعة SubVIs ودوال الى مدخل نفس الـ SubVI او الدالة او مجموعة SubVIs والدوال.

فكما نعلم من مفهوم Dataflow لن نستطيع وصل مخرج الى مدخل لنفس الدالة او SubVI وذلك لانها لن تنفذ و تخرج بيانات على مخارجها الا اذا كانت البيانات جاهزة على مداخلها وهو لن يحدث لان احد المدخل موصل باحد المخارج لها.



ففى هذة الحالة نستخدم Feedback Node لكى نجعل القيمة التى خرجت على احد المخارج لدالة او SubVI هى القيمة الداخلة على احد المداخل لتلك الدالة او SubVI فى المرة التالية التى تنفذ فيها هذة الدالة او SubVI.



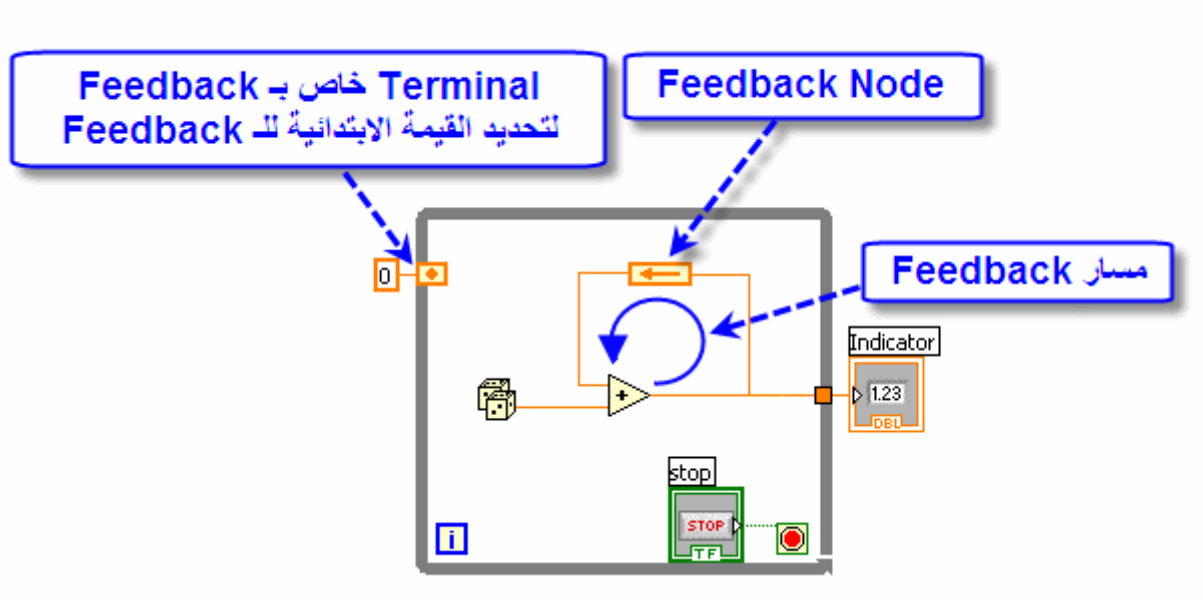
وهى مثل Shift Register تحتفظ بقيمتها بعد انتهاء التنفيذة الحالية لتنقلها الى التنفيذة التالية.

و يمكن ان تنقل اى نوع من البيانات.

ويوجد Terminal مع Feedback Node هذا Terminal يوصل اليه القيمة الابتدائية للـ Feedback Node.

هذة القيمة الابتدائية ستكون هى قيمة المدخل الموصل الية Feedback Node فى اول مرة تنفذ فيها Loop.

وهذا Terminal يوضع تلقائيا بمجرد ادارج Feedback Node.

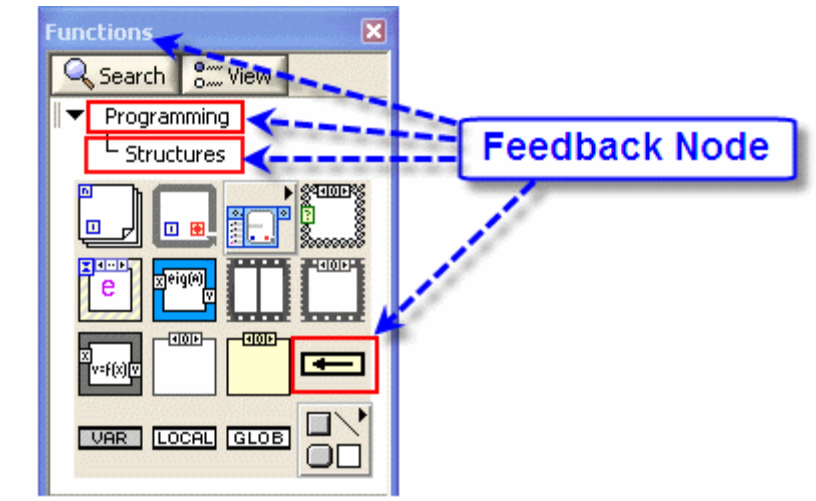


وكما بالشكل يحدد اتجاه السهم الموجود فى Feedback Node مسار تدفق البيانات.

ادراج Feedback Node :

يتم ادراج Feedback Node من

Function Palette>>Programming>>Structures

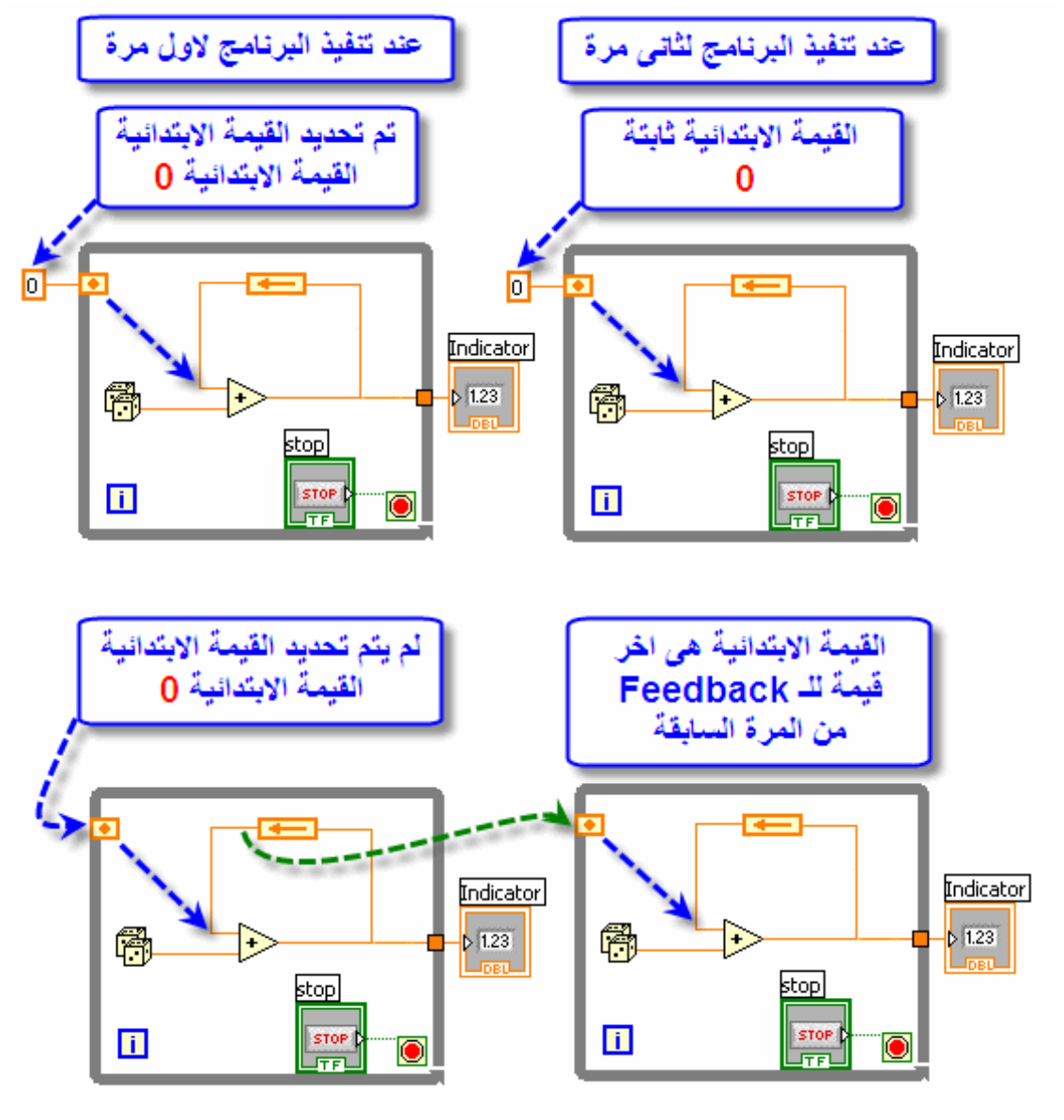


ماذا يحدث اذا لم يتم تحديد القيم الابتدائية للـ Feedback Node؟

اذا قمت بتحديد القيمة الابتدائية للـ Feedback Node فهذة القيمة ستكون قيمة Feedback Node قبل الدخول فى التنفيذة الاولى فى Loop وهذا سواء كان تنفيذ البرنامج لاول مرة او ثانى مرة او اى مرة.

انما اذا لم يتم تحديد القيمة الالبتدائية للـ Feedback Node سوف يأخذ Feedback Node القيمة الابتدائة الخاصة بنوع البيانات التى تنقلها وذلك قبل اول مرة تنفذ فيها Loop ولكن سوف يحتفظ Feedback Node باخر قيمة له بعد انتهاء Loop لتكون تلك القيمة هى القيمة الابتدائية للـ Feedback Node فى المرةالتالية التى تنفذ فيها Loop وهكذا.

وهذا مماثل لما يحدث فى Shift Registers.



Case Structure

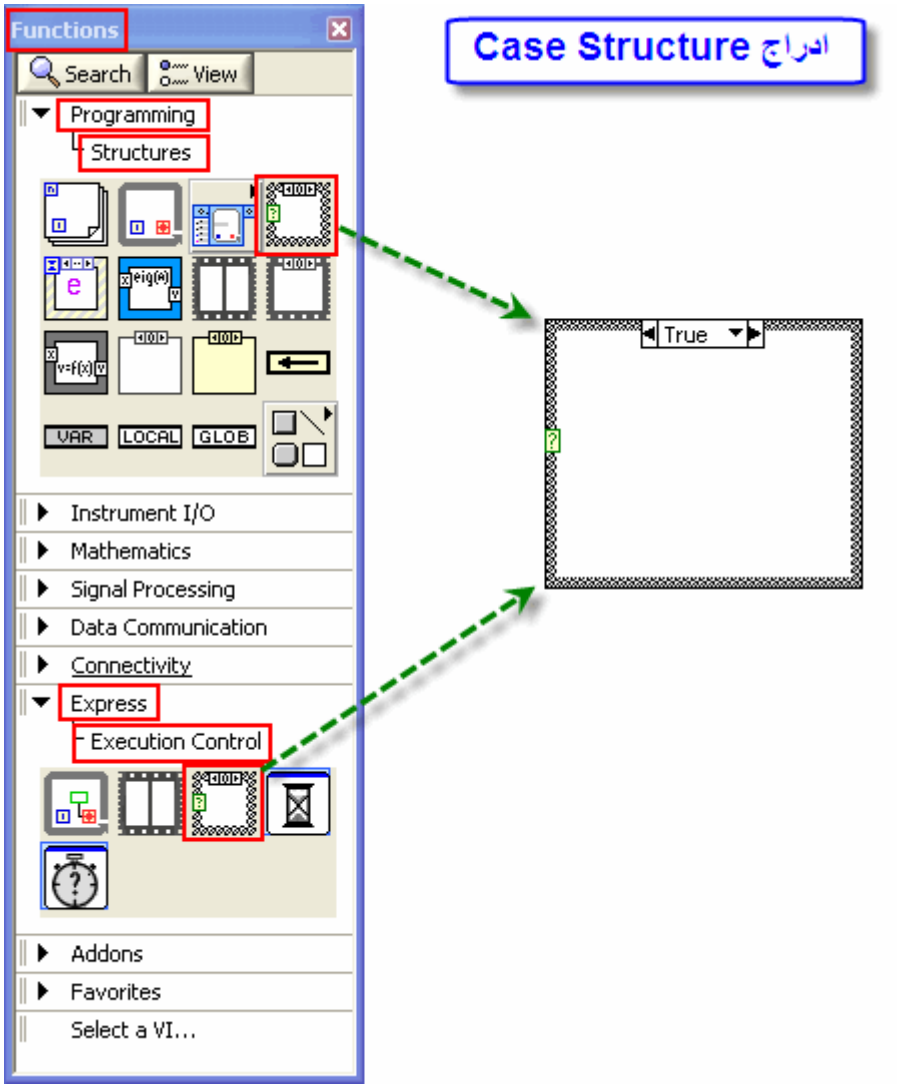
هى احد الطرق لتنفيذ الجمل الشرطية وهى تعادل If….then…else فى لغات البرمجة الاخرى.

ويتم ادراج Case Structure من

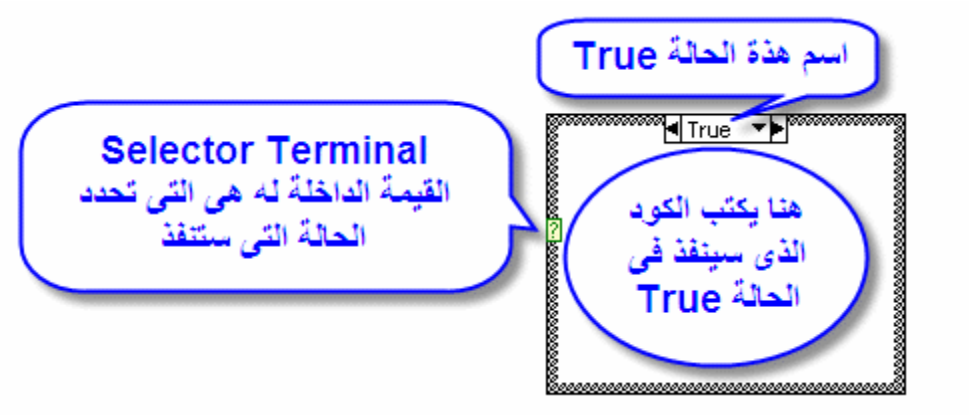
Function Palette>>Programming>>Structures

او من

Function Palette>>Express >>Execution Control

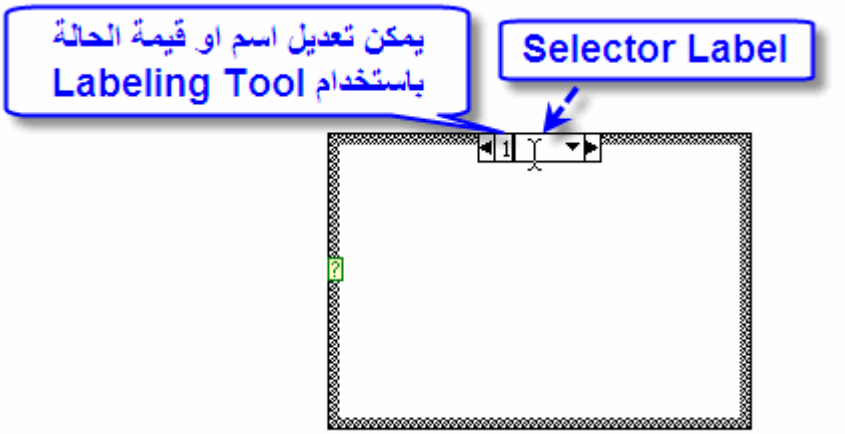


وتتكون Case Structure من عدة حالات ومدخل ⍰ يسمى Selector Terminal .وكل حالة يكتب فيها كود هذا الكود يسمى Subdiagram.وعند تنفيذ Case Structure يتم تنفيذ حالة واحدة وتحدد القيمة التى يأخذها المدخل الحالة التى ستنفذ.



ولكل حالة اسم وهو القيمة التى ستنفذ عندها الحالة.

ويكتب اسم الحالة فى Selector Label ويمكن التعديل فيه باستخدام Labeling Tool.



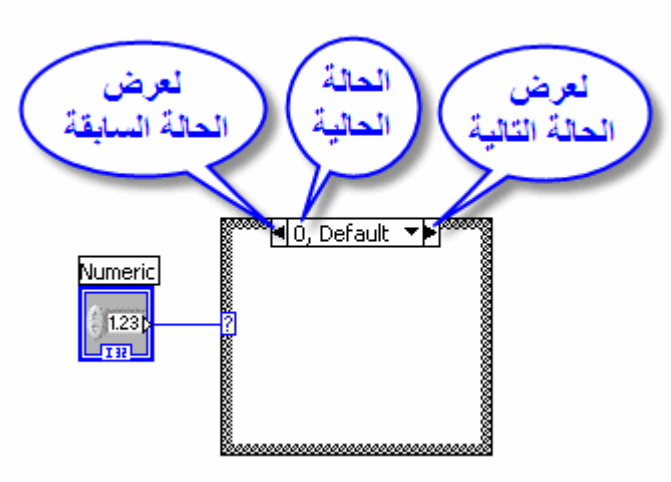
عادة ما تحتوى Case Structureعلى عدة حالات ولكن يعرض محتوى حالة واحدة

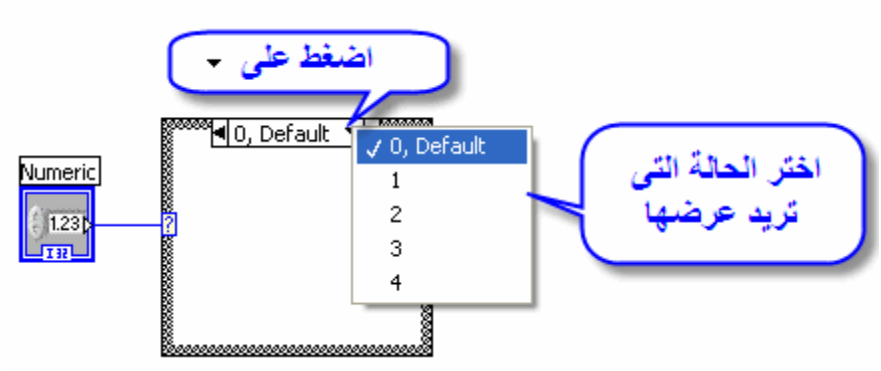
ولعرض محتويات الحالة التالية نستخدم ▶ السهم الايمن

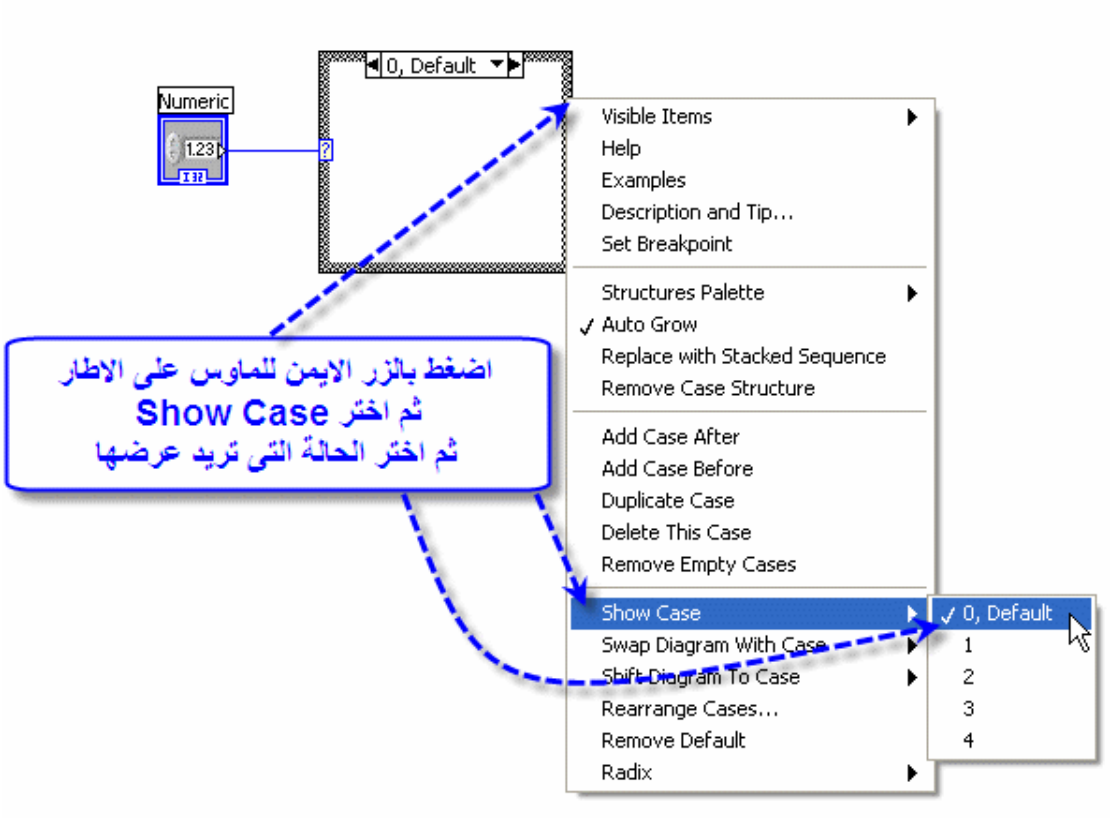
و لعرض محتويات الحالة السابقة نستخدم ◀ السهم الايسر.

ويمكن التنقل الى اى حالة وذلك بالضغط السهم ▾و اختيار الحالة التى يراد عرضها.

اوبالضغط بالزر الايمن للماوس على اطار Case واختيار Show Case واختيار الحالة التى تريدها

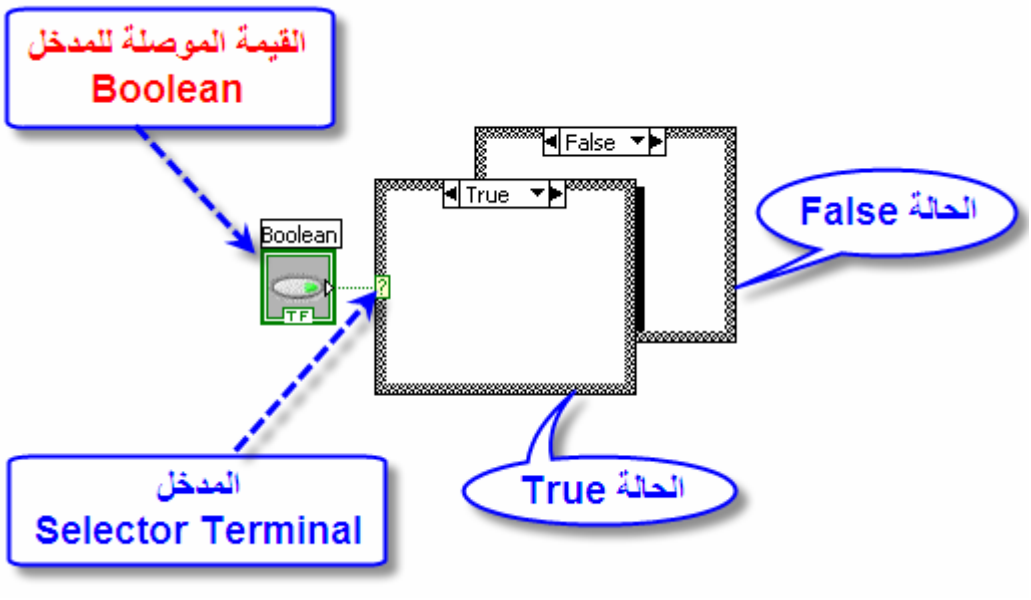

لعرض
الحالة السابقة
الحالة
الحالية
لعرض
الحالة التالية
0, Default
Numeric
1.23


اضغط على
0, Default
Numeric
1.23
✓ 0, Default
1
2
3
4
اختر الحالة التى
تريد عرضها


0, Default
Numeric
1.23
Visible Items
Help
Examples
Description and Tip...
Set Breakpoint
Structures Palette
✓ Auto Grow
Replace with Stacked Sequence
Remove Case Structure
Add Case After
Add Case Before
Duplicate Case
Delete This Case
Remove Empty Cases
Show Case
Swap Diagram With Case
Shift Diagram To Case
Rearrange Cases...
Remove Default
Radix
✓ 0, Default
1
2
3
4
اضغط بالزر الايمن للماوس على الاطار
ثم اختر Show Case
ثم اختر الحالة التى تريد عرضها

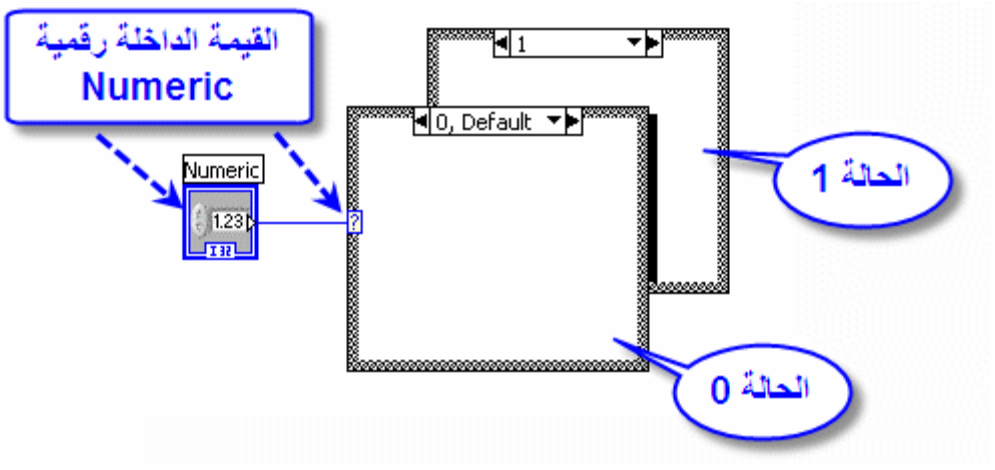
كيفية تحديد الحالات مع الانواع المختلفة للقيم الموصلة للـ Selector Terminal:

- اذا اكان دخل المدخل من النوع Boolean فان هناك حالتين هما True او False

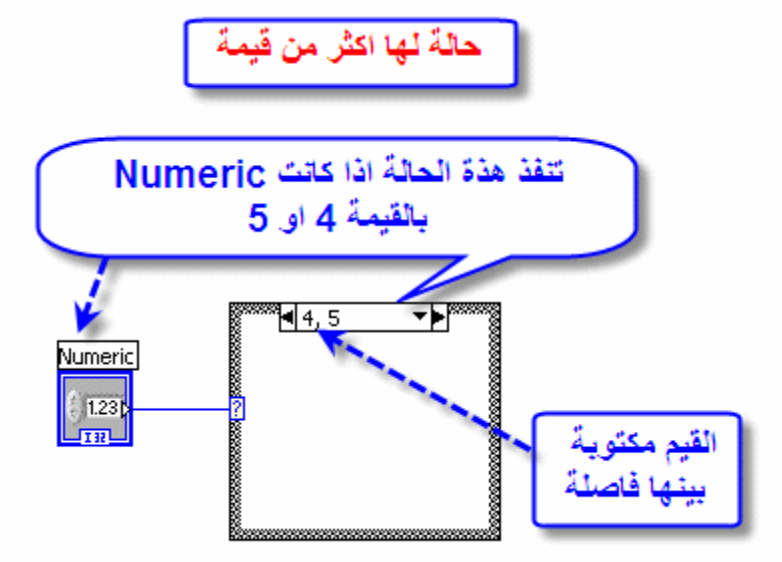


- اذا كان دخل المدخل من النوع Numeric (رقمى) فيمكن ان يوجد عدد نهائى من القيم.

وفى هذة الحالة تحتوى Case Structure على حالتين فقط ويمكن ان تضيف اليها الحالات التى تريدها.



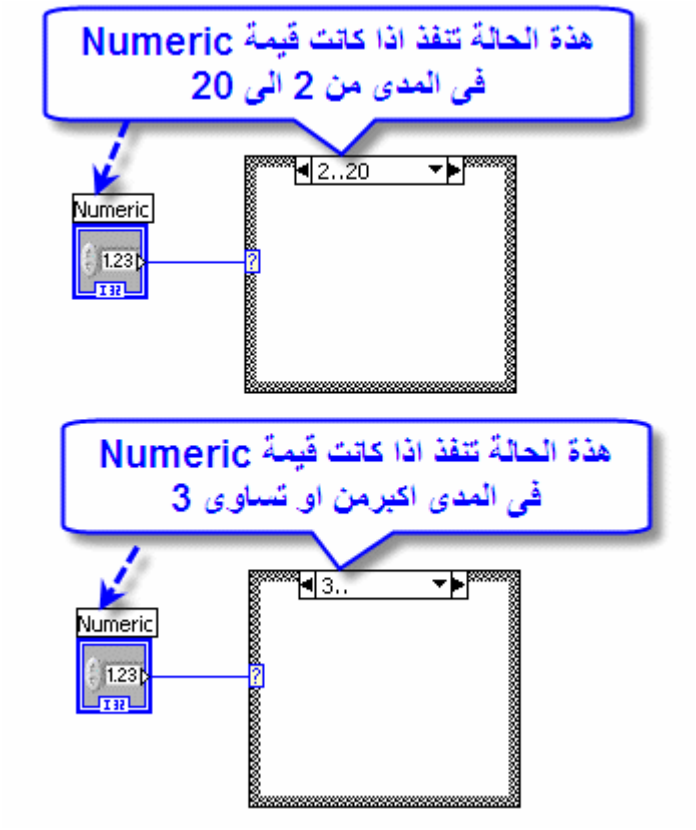
يمكن ان تكون للحالة اكثر من قيمة ولعمل ذلك تكتب القيم بينها فاصلة .



كما يمكن ان تكون الحالة لمدى من القيم ولعمل ذلك نستخدم النقطتين .. للتعبير عن المدى

فمثلا 2..20 تعنى ان هذة الحالة لكل الارقام الصحيحة من 2 الى 20.

و ..3 تعنى ان الحالة لكل الارقام الصحيحة التى تساوى اواكبر من 3.



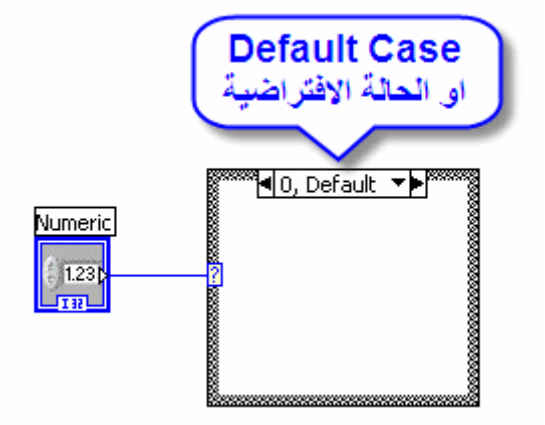
Default Case او الحالة الافتراضية :

هى الحالة التى ستنفذ اذا كانت القيمة الموصلة للمدخل ليس مخصص لها حالة .

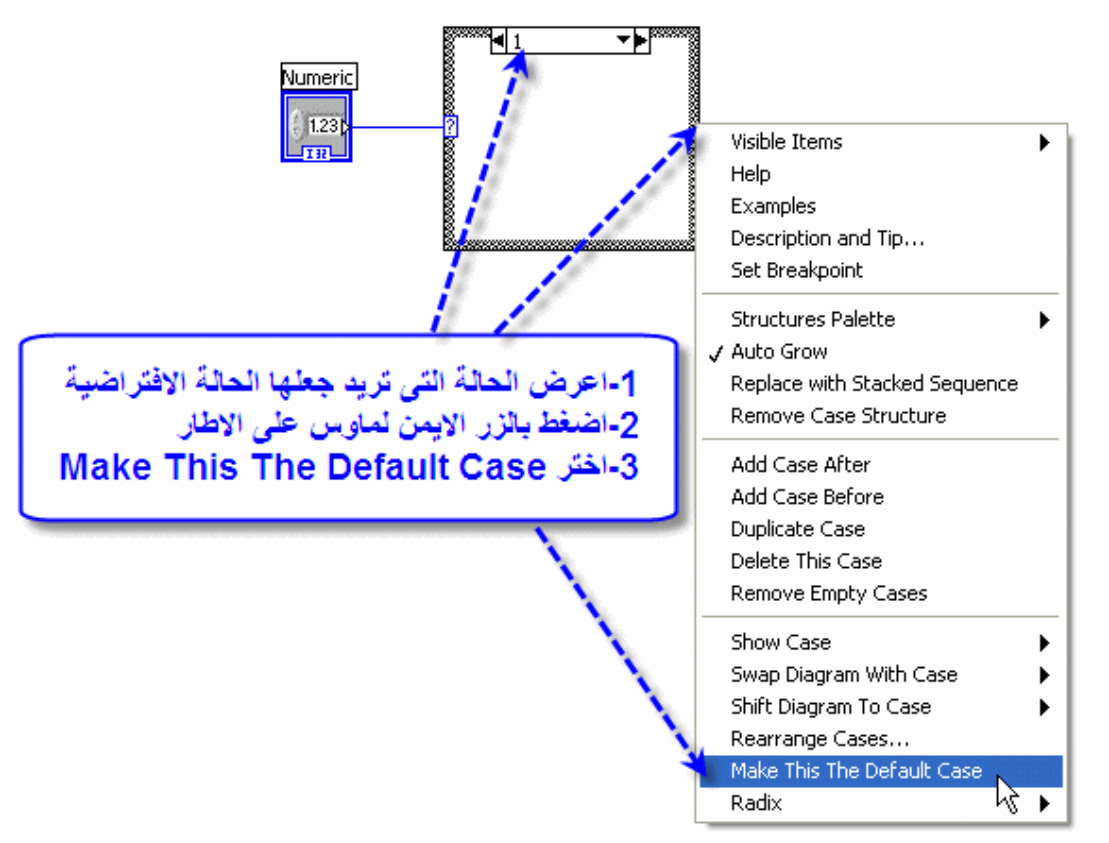
مثال اذا كانت الحالات الموجودة هى للقيم 1و2و3 فقط فما هى الحالة التى ستنفذ اذا كان قيمة الدخل 100 مثلا.

ستنفذ الحالة الافتراضية (Default Case)

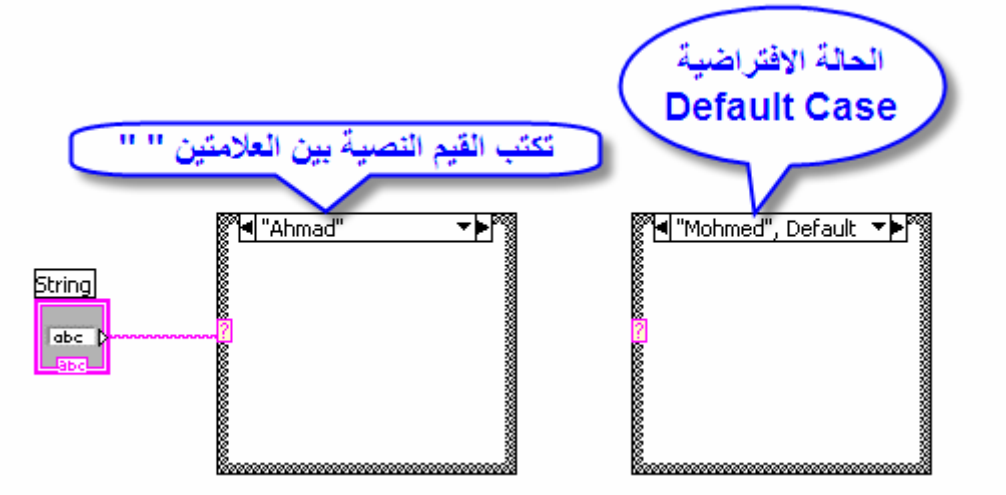
و الحالة الافتراضية يظهر بجوار القيمة المخصصة لها كلمة Default .



ويمكن جعل اى حالة من الحالات هى الحالة الافتراضية (Default Case).

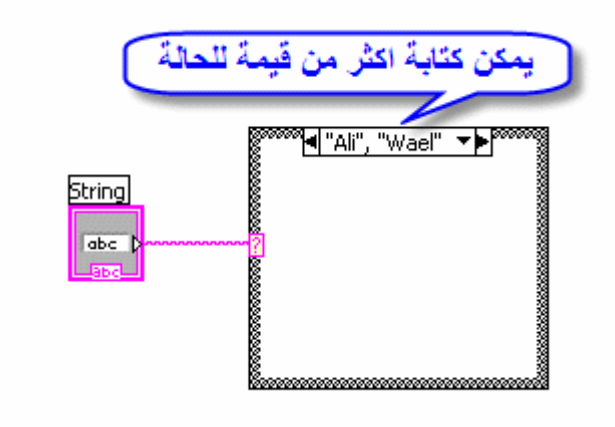


- اذا كان دخل المدخل من النوع String (نص) فتكتب قيم الحالات بين علامتين تنصيص واذا لم تضع العلامتين فسوف يقوم LabVIEW بوضعهم تلقائيا .



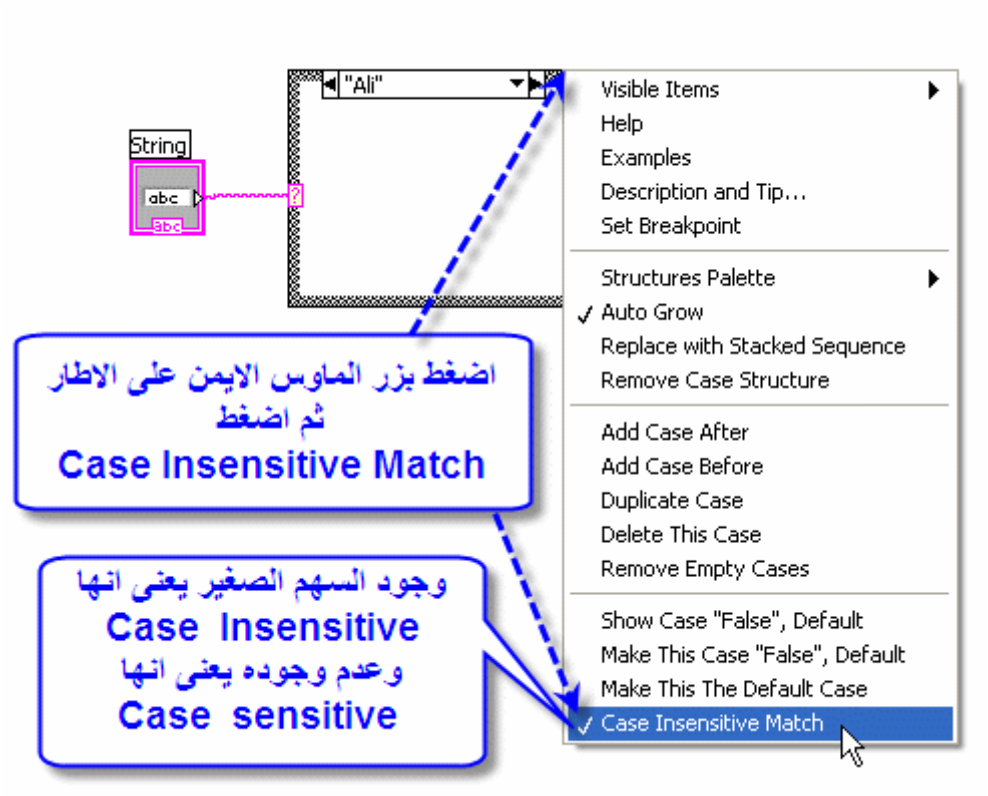
لاحظ ان كلمة Default لا توجد بين علامتين تنصيص.

يمكن ان تكون للحالة اكثر من قيمة وذلك بكتابة القيم بينها فاصلة.



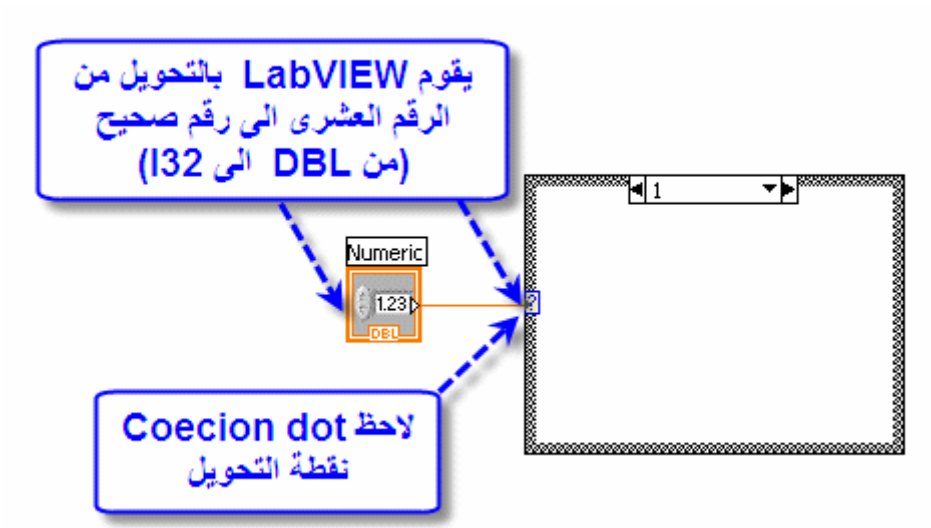
يمكن ان نجعل الدخل Case Sensitive او Case Insensitive.

- Case Sensitive تعنى الاخذ فى الاعتبار الحروف الكبيرة فالحروف الصغيرة مثلا ALI لا يطابق ali.
- Case Insensitive تعنى انه لافرق بين الحروف الكبيرة والحروف الصغيرة فمثلا ALI تطابق Ali تطابق ali.



لاحظ انه:

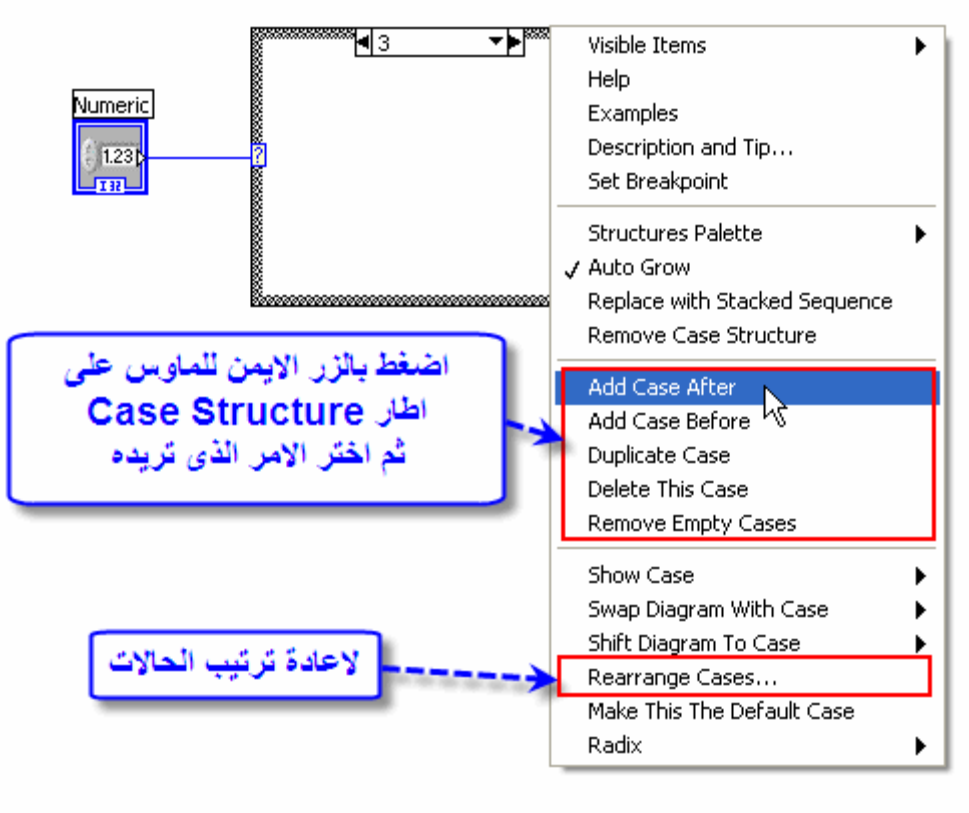
- اذا تم توصيل رقم من النوع Floating-Point فسوف يقوم LabVEW بتحويل القيمة الى رقم صحيح من النوع I32 ثم يختار الحالة على اساس الرقم المحول.



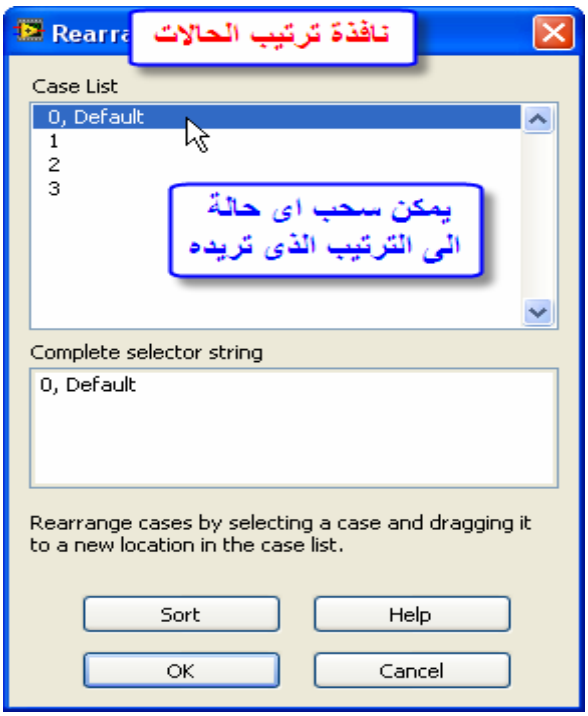
- يمكن نقل مكان المدخل (Case Selector) الى اى مكان على اطار Case Structure.

اضافة وحذف حالات :

يمكن اضافة حالة جديدة او حذف حالة او عمل نسخة من حالة موجودة و خيارات اخرى وذلك بالضغط بالزر الايمن للماوس على اطار Case Structure.



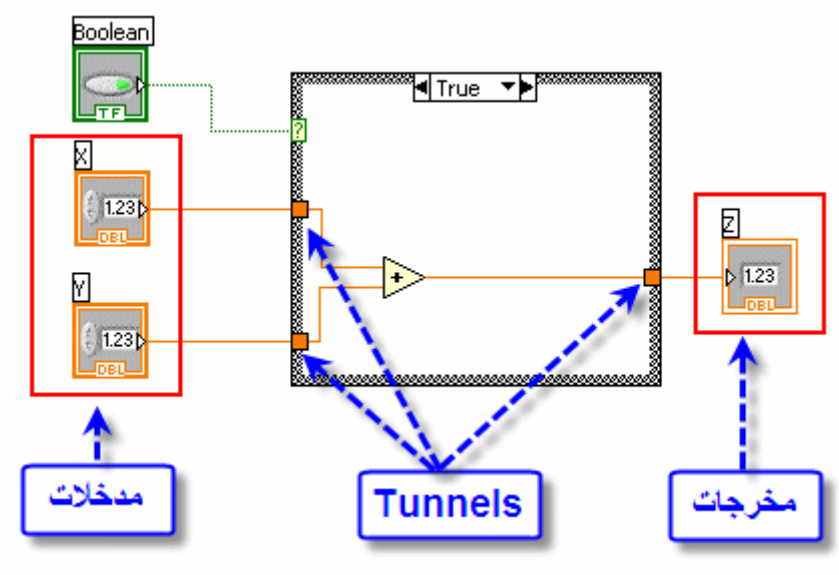
وعند اختيار اعادة ترتيب الحالات تظهر لنا نافذة نحدد فيها ترتيب الحالات



الادخال و الاخراج من و الى Case Structure :

ذكرنا ان هناك المدخل (Selector Terminal) وقيمته هى التى تحدد الحالة التى ستنفذ.

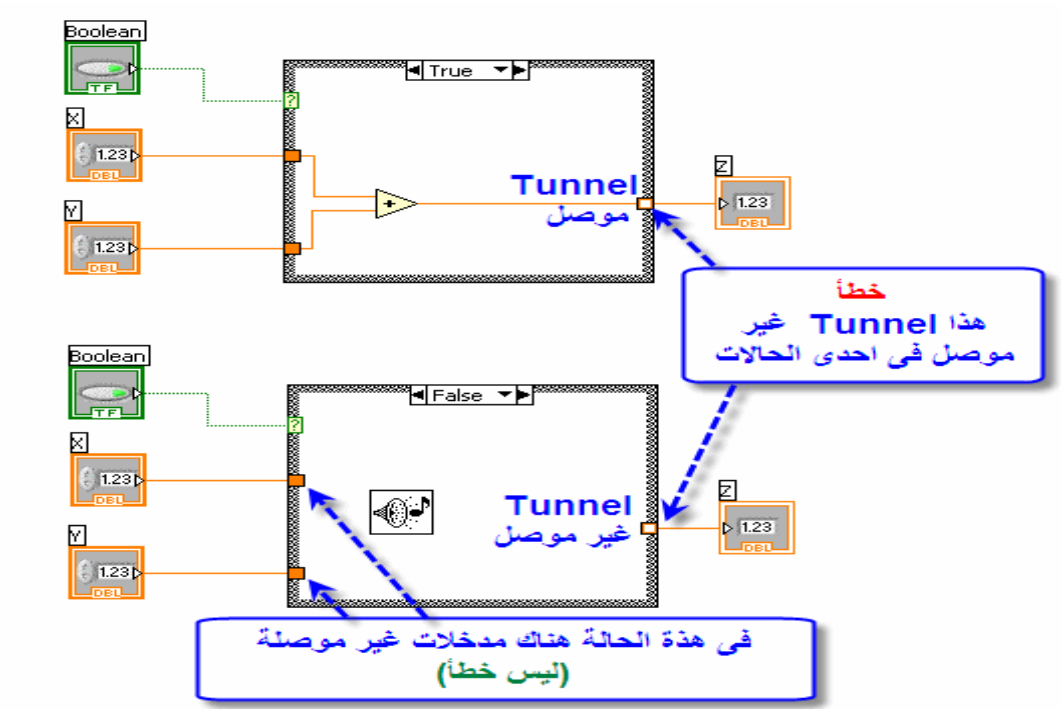
يمكن ايضا ان ندخل اى عدد من المدخلات الى الحالات او نخرج اى عدد من المخرجات و ذلك من خلال Tunnels .



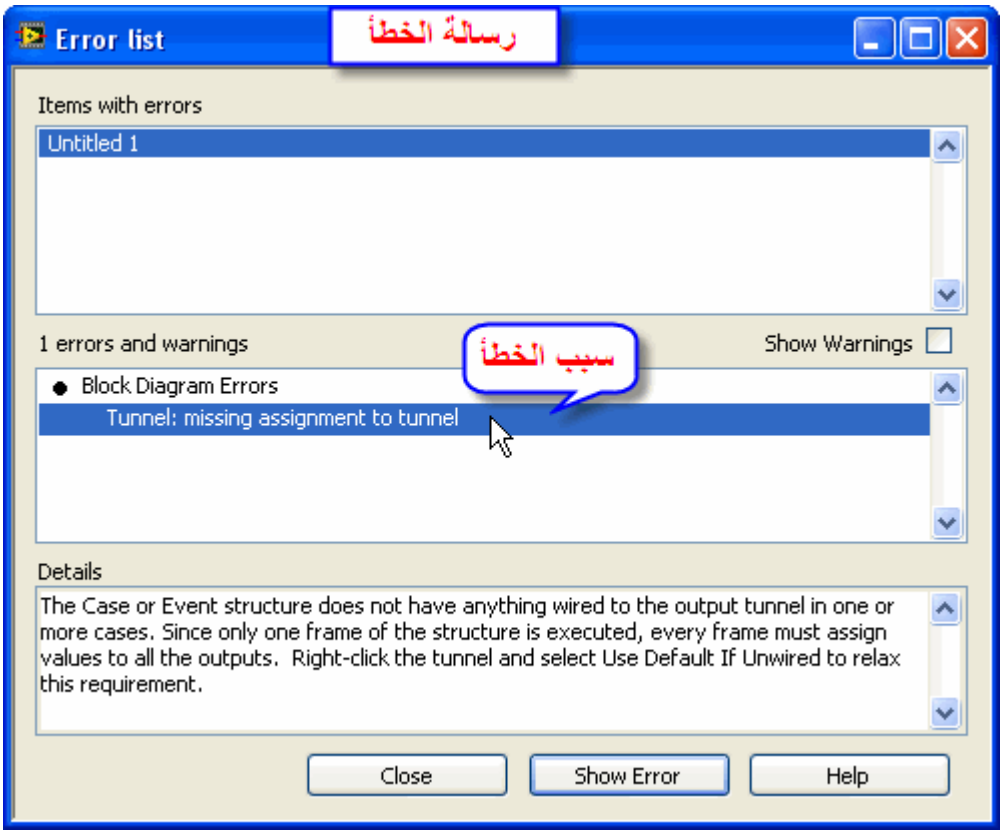
ليس شرطا ان يكون للحالات مدخلات ومخرجات.

وليس شرطا ان تستخدم الحالة كل المدخلات .

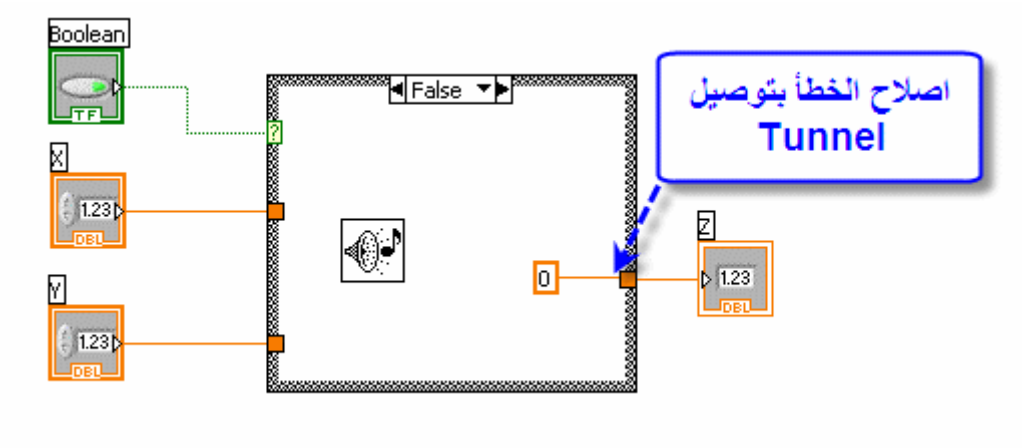
ولكن اذا تم توصيل مخرج فى اى حالة فلابد ان يوصل هذا المخرج فى جميع الحالات .والا لن ينفذ البرنامج وسوف يعطى لك رسالة خطأ تفيد بانه هناك Tunnel لمخرج غير موصل.



فى هذة الحالة عند محاوله تنفيذ البرنامج تظهر لنا هذة الرسالة



ولاصلاح هذا الخطأ نقوم بالاتى:



Dialogs

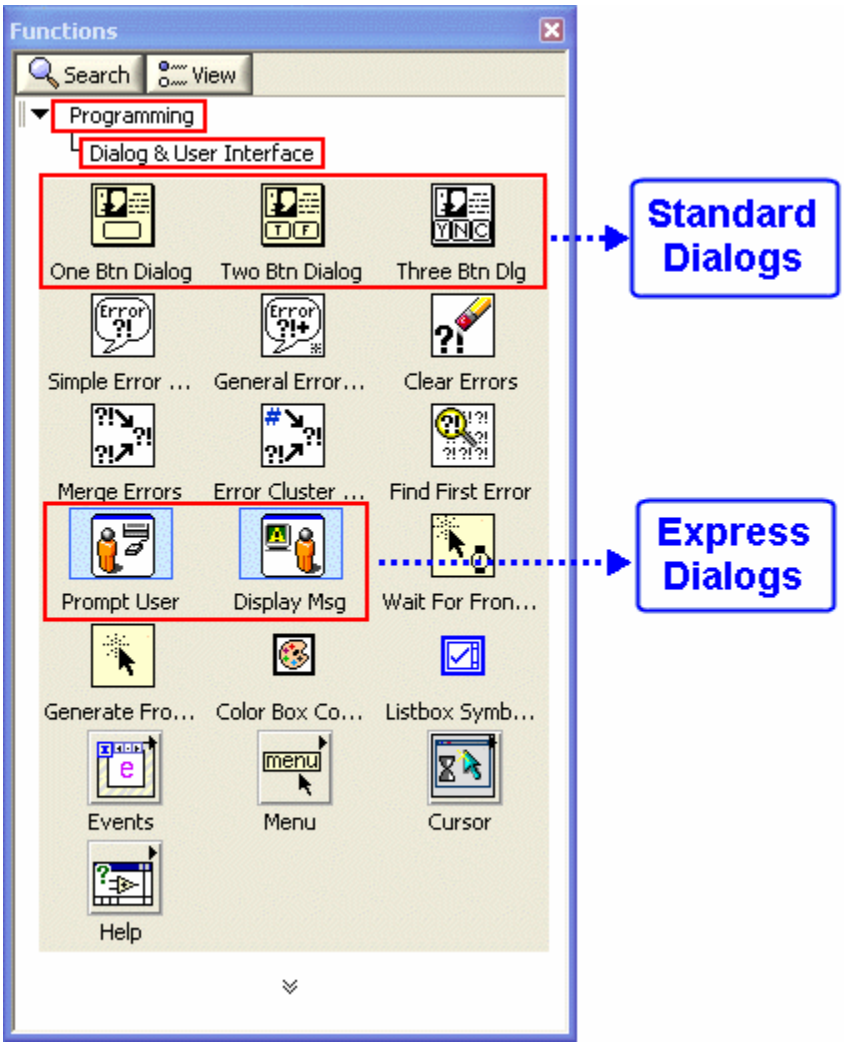
كيف تجعل البرنامج يتفاعل مع المستخدم ؟

سوف نتوقف قليلا للتعرف عن كيف اخراج رسالة للمستخدم اثناء تنفيذ البرنامج.وهذة الرسائل يمكن ان تحتوى على اكثر من زرار مثل زرار Ok او Cancel.

ويحتوى LabVIEW على بعض الادوات التى تتيح اظهار رسائل اثناء تنفيذ البرنامج.

ويمكن ادراج هذة الادوات من

Function Palette>>Programming>>Dialog & User Interface



ويوجد نوعان من الادوات التى يتيحها LabVIEW لاظهار الرسائل :

Standard Dialogs-1 :

وهى دوال لاظهار رسائل بها نص وهذة الرسائل قد يوجد بها زرار او زرارين او ثلاث زراير وذلك حسب الدالة المستخدمة.

ومدخلات هذة الدوال تكون عبارة عن :

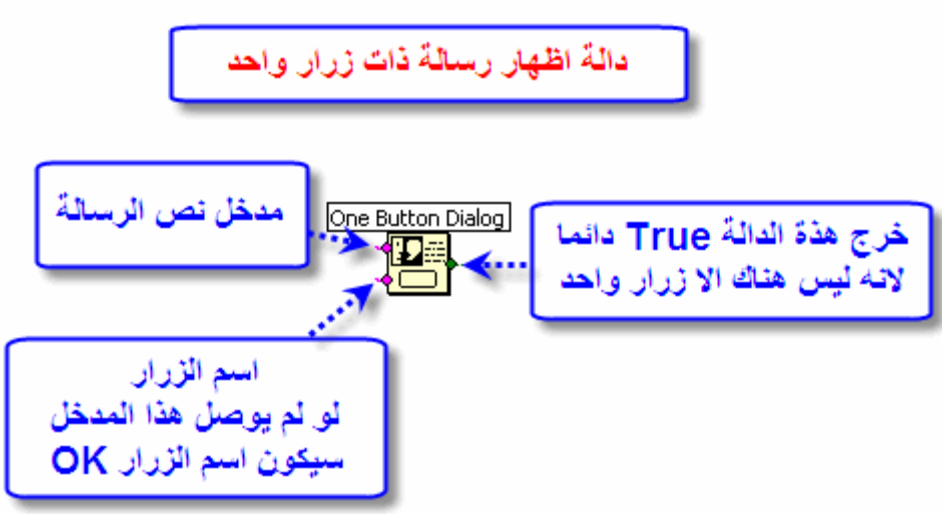
- نص الرسالة التى تريد اظهارها للمستخدم اثناء تنفيذ البرنامج .
- اسماء الزراير التى ستوجد فى الرسالة مثل Cancel او Ok.

اما المخرجات فتحدد اى زرار قد ضغط عليه المستخدم حتى يتم على اساسه تحديد مسار تنفيذ البرنامج.

وعدد هذة الدوال ثلاث هم:

One Button Dialog

وهى لاظهار رسالة ذات زرار واحد.



مثال

Two Button Dialog

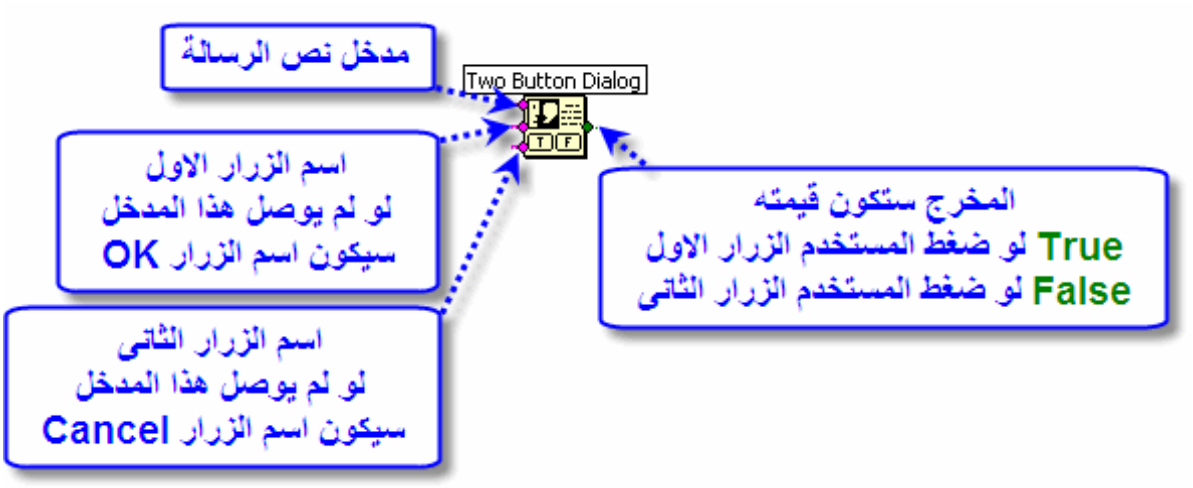
وهى لاظهار رسالة ذات زرارين عادة يكونان زرارى OK و Cancel.

ويوجد لهذة الدالة ثلاث مداخل :

- مدخل لنص الرسالة
- مدخل لاسم الزرار الاول وهذا زرار True لان الدالة ستخرج القيمة True اذا ضغط المستخدم هذا الزرار.

  واذا لم يوصل هذا المدخل سيكون اسم الزرار OK .
- مدخل لاسم الزرار الثانى ويسمى زرار False لان الدالة ستخرج القيمة False اذا ضغط المستخدم هذا الزرار.

  واذا لم يوصل هذا المدخل سيكون اسم الزرار Cancel.

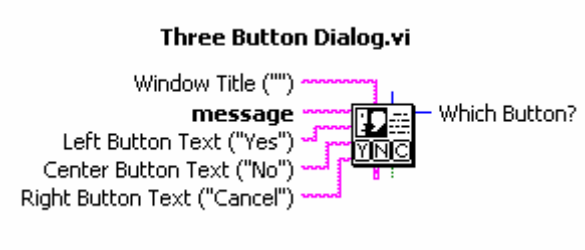


مثال

Two Button Dialog

لاظهار رسالة ذات ثلاث زراير عادة Yes و No و Cancel.

ويوجد لهذة الدالة عدة مداخل ولكن اهم هذة المداخل كما فى الدالتين السابقتين:



- مدخل نص الرسالة
- مدخل لاسم الزرار الاول وسيظهر هذا الزرار فى يسار الرسالة واذا لم يوصل هذا المدخل سيكون اسم الزرار Yes .
- مدخل لاسم الزرار الثانى وسيظهر هذا الزرار فى وسط الرسالة واذا لم يوصل هذا المدخل سيكون اسم الزرار NO .
- مدخل لاسم الزرار الثالث وسيظهر هذا الزرار فى يمين الرسالة واذا لم يوصل هذا المدخل سيكون اسم الزرار Cancel .

ويوجد لهذة الدالة مخرج واحد يحدد اى زرار تم الضغط عليه.

وقيمة هذا المخرج ستكون بالقيمة :

0 اذا ضغط المستخدم الزرار الاول.

1 اذا ضغط المستخدم الزرار الثانى.

2 اذا ضغط المستخدم الزرار الثالث.

3 اذا اغلق المستخدم الرسالة دون الضغط على اى من الزراير.

2-Express Dialogs :

وهى Express VI سهلة الاستخدام وسريعة لاظهار رسائل للمستخدم تخبره بشىء معين او تطلب منه ادخال بيانات معينة.

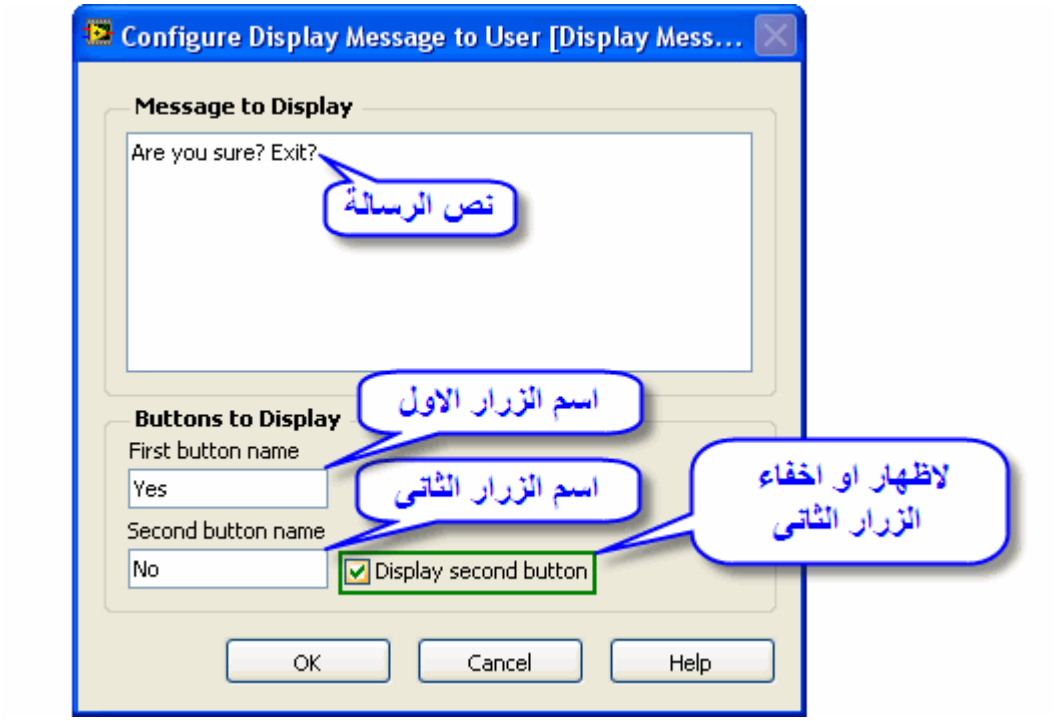
وهما Display Message Express VI و Prompt User Express VI.

Display Message Express VI

تستخدم هذة Express VI لاظهار رسالة للمستخدم بها زرار او زرارين.

هذة Express VI تشبة الدوال الاساسية One and Two Button Dialogs.

بمجرد ادراج Express VI فى Block Diagram تظهر لنا نافذة يتم تحديد فيها خائص الرسالة التى نريدها



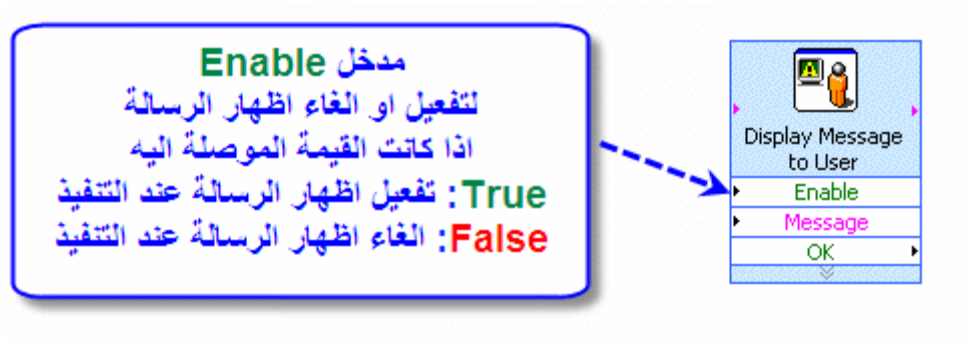
وبعد ادراج هذة Express VI يمكن ان نعيد فتح نافذة الخصائص وذلك بالضغط ضغطتين متتاليتين او بالضغط عليها بالزر الايمن و اختيار Properties. ExpressVI

ويوجد لهذة Express VI مدخل يسمى Enabled وهو لتفعيل او عدم تفعيل اظهار الرسالة فاذا كان كانت القيمة الموصلة لهذا المدخل :

True سوف يتم اظهار الرسالة عند تنفيذ Express VI.

False لن يتم اظهار الرسالة عند تنفيذ Express VI.

وهذا للتحكم فى اظهار او عدم اظهار الرسالة من داخل البرنامج.



ويوجد لهذة Express VI مخرج قيمته تتوقف على الزرار الذى تم الضغط عليه من قبل المستخدم عند اظهار الرسالة.

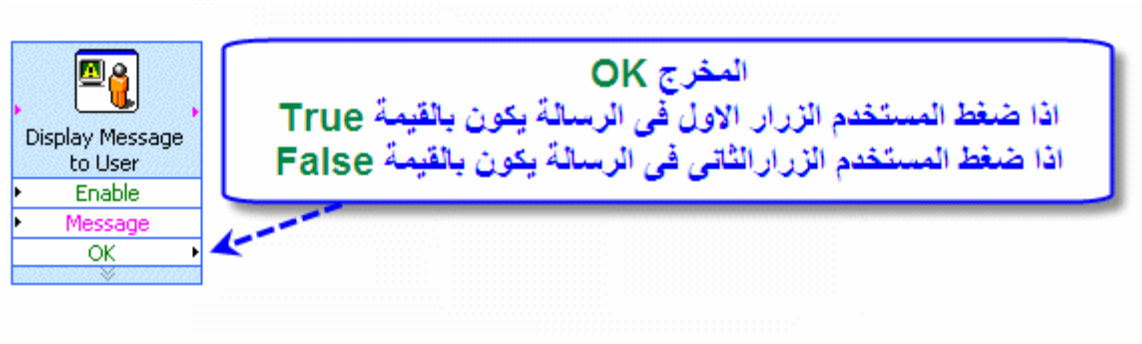
ففى حالة اذا كانت الرسالة تحتوى على زرار واحد فان هذا المخرج قيمته True دائما.

اما فى حالة اذا كانت الرسالة تحتوى على زرارين فستكون قيمته :

True اذا تم الضغط على الزرار الاول.

False اذا تم الضغط على الزرار الثانى.

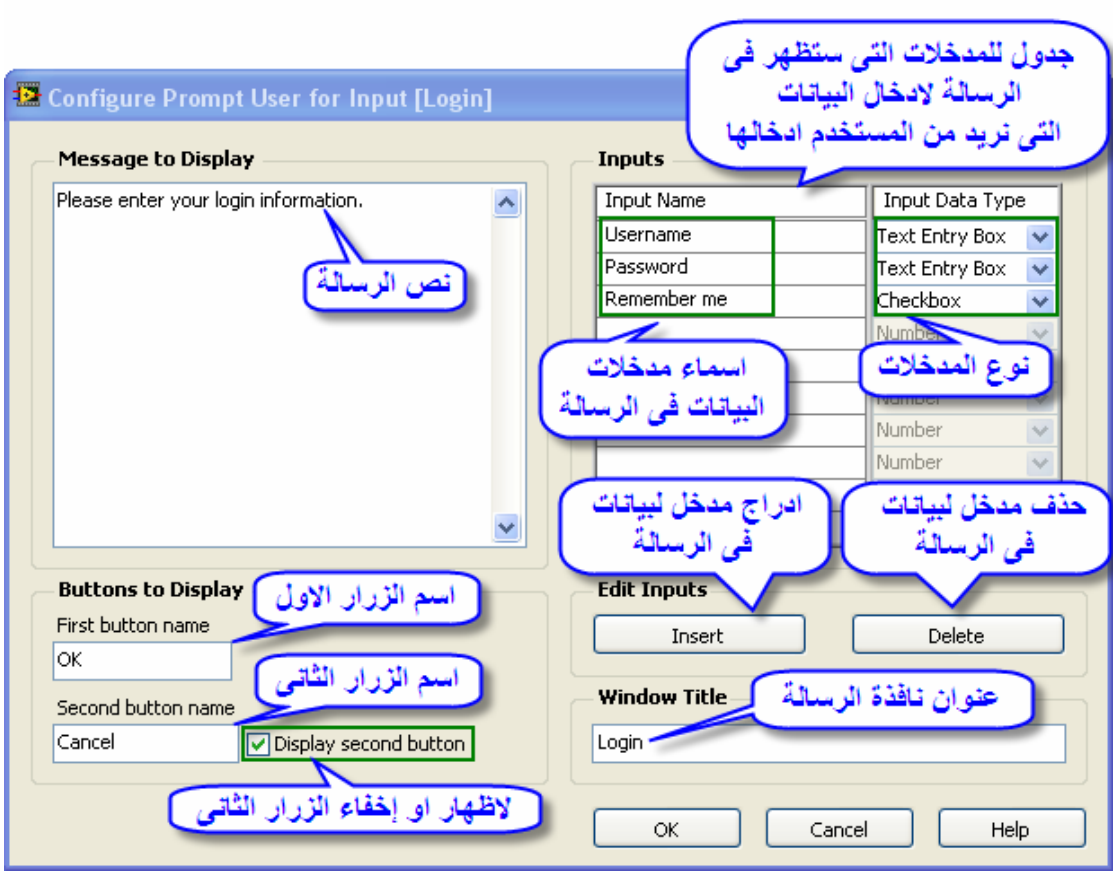
وهذا يماثل الدالتين One Button Dialog و Two Button Dialog



Prompt User Express VI

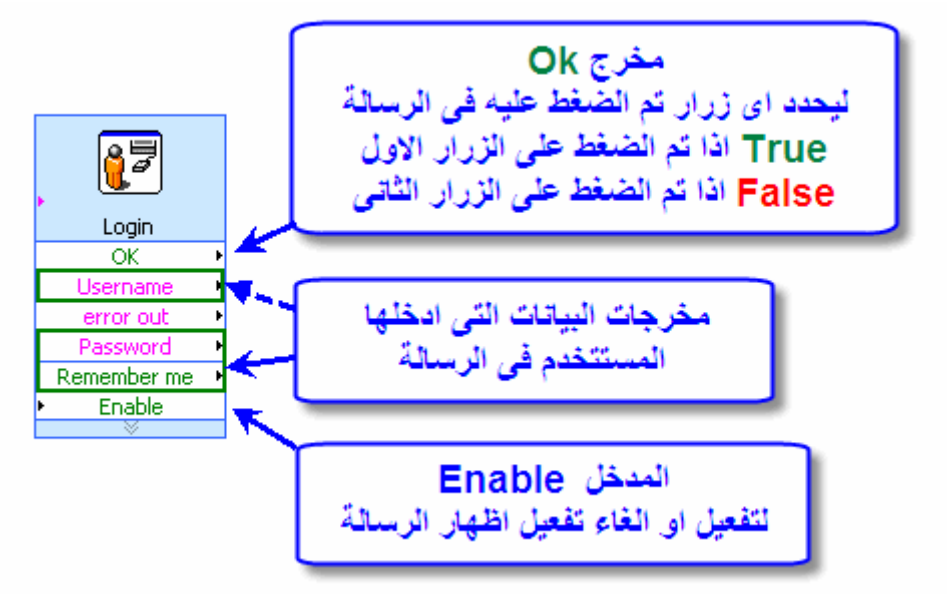
وهى لاظهار رسالة يمكن للمستخدم من خلالها ادخال بيانات محددة للبرنامج مثل رسالة ادخال اسم المستخدم وكلمة السر.

وبمجرد ادراج هذة Express VI تظهر لنا نافذة الخصائص لتحديد خصائص الرسالة.



شكل الرسالة

شكل ايقونة Express VI بعد تحديد خصائصها



The Sequence Structure

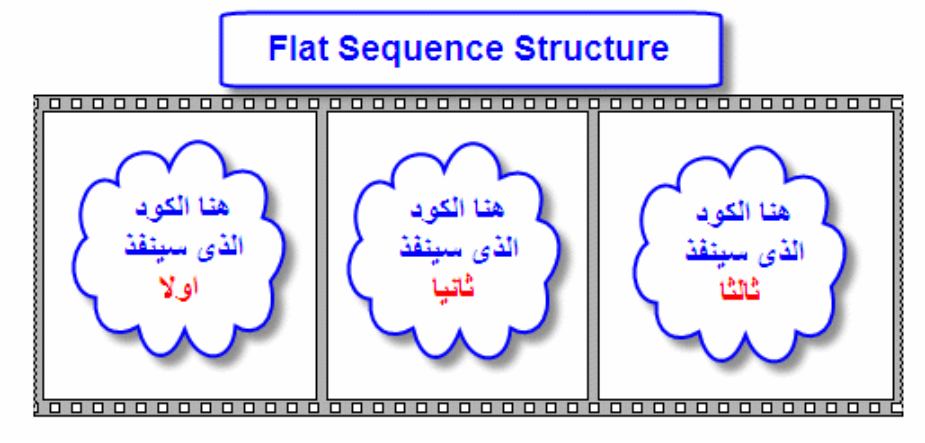
تحتوى Sequence Structure مجموعة من الاكواد كل كود يسمى SubDiagram تنفذ كلها بترتيب تواجدها فى Sequence Structure.ويوضع كل كود فى اطار. ترتيب هذا الاطار هو ترتيب تنفيذ الكود.

ويوجد نوعان من Sequence Structure هما Flat Sequence Structure و Stacked Sequence Structure.

وهما يختلفان فقط فى طريقة عرض الاكواد (SubDiagrams) .

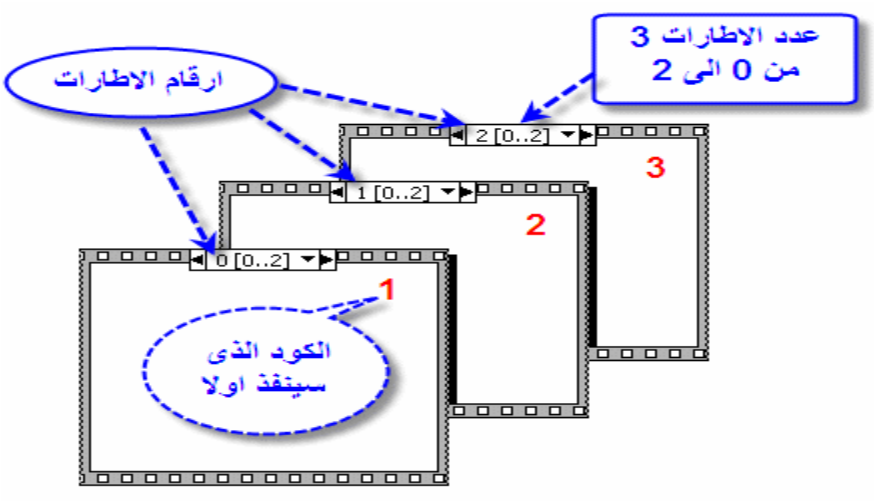
Flat Sequence Structure

تعرض الاطارات او الاكواد بجوار بعضها وتنفذ الاكواد الموجدة داخل الاطارت بالترتيب من اليسار الى اليمين.



Stacked Sequence Structure

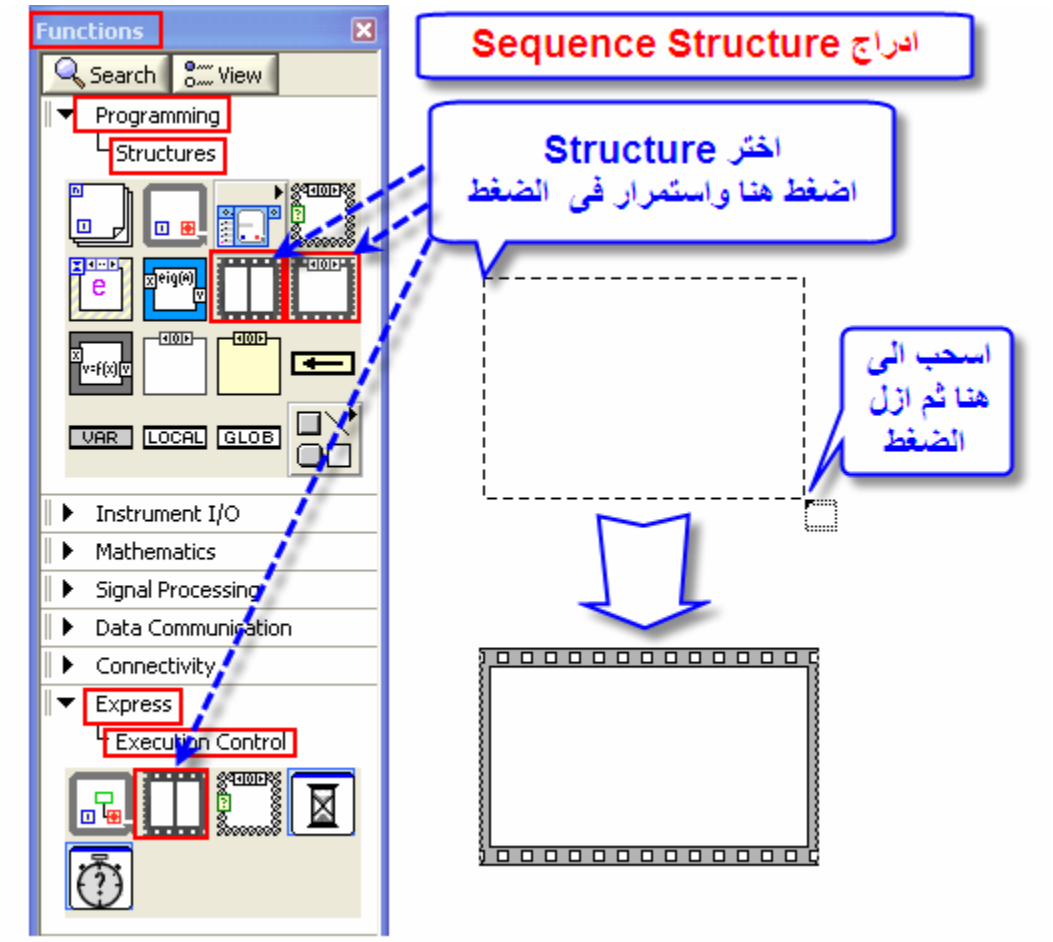
وهى تماثل Case Structure فى طريقة عرض الاكواد(الموجودة فى كل اطار).



ادراج Sequence Structure :

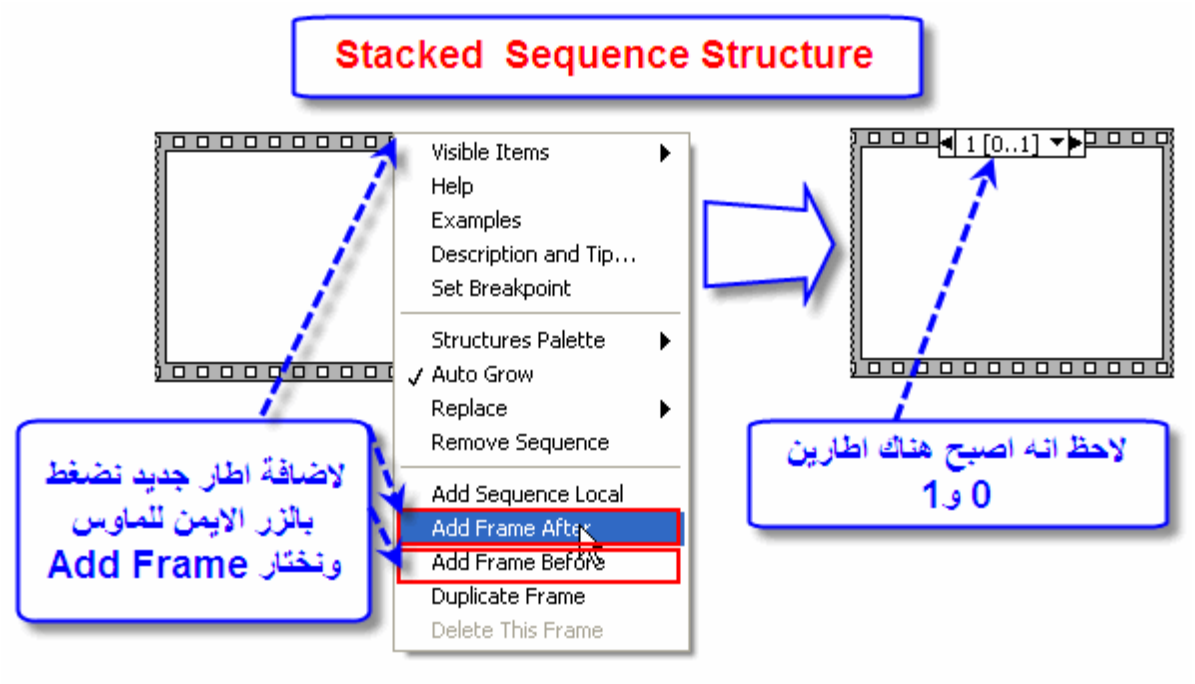
يتم ادراج Sequence Structure من

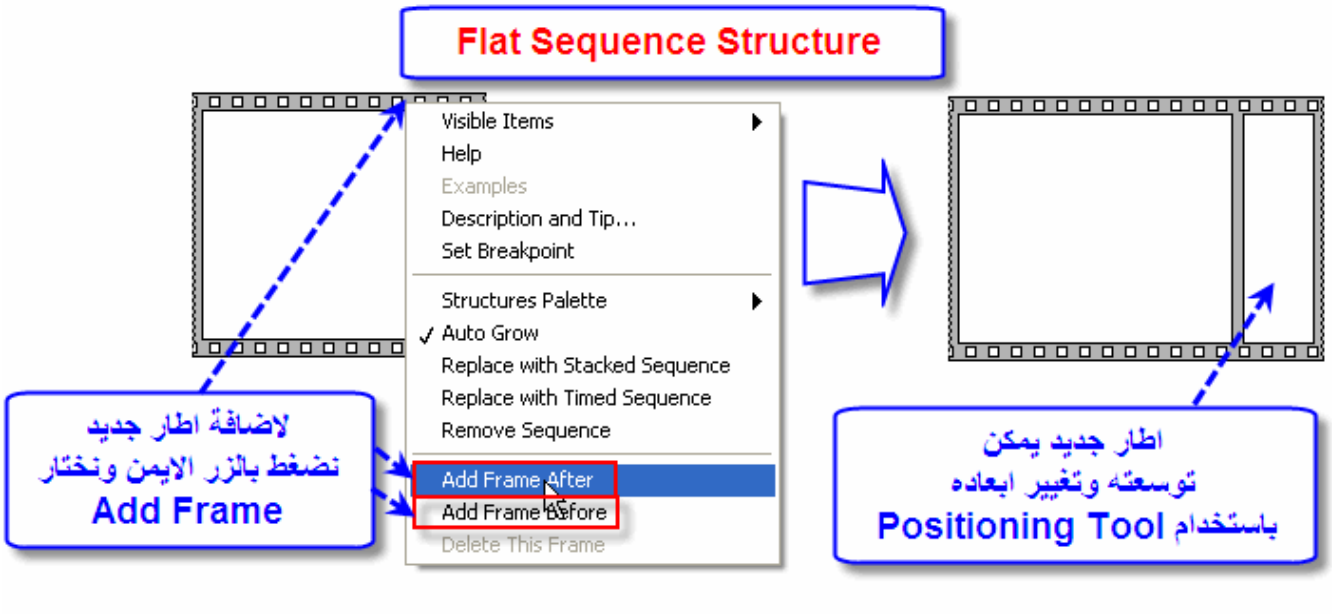
Function Palette>>Programming>>Structures



ولاضافة اطار جديد نضغط بالزر الايمن اطار Structure ونضغط Add Frame

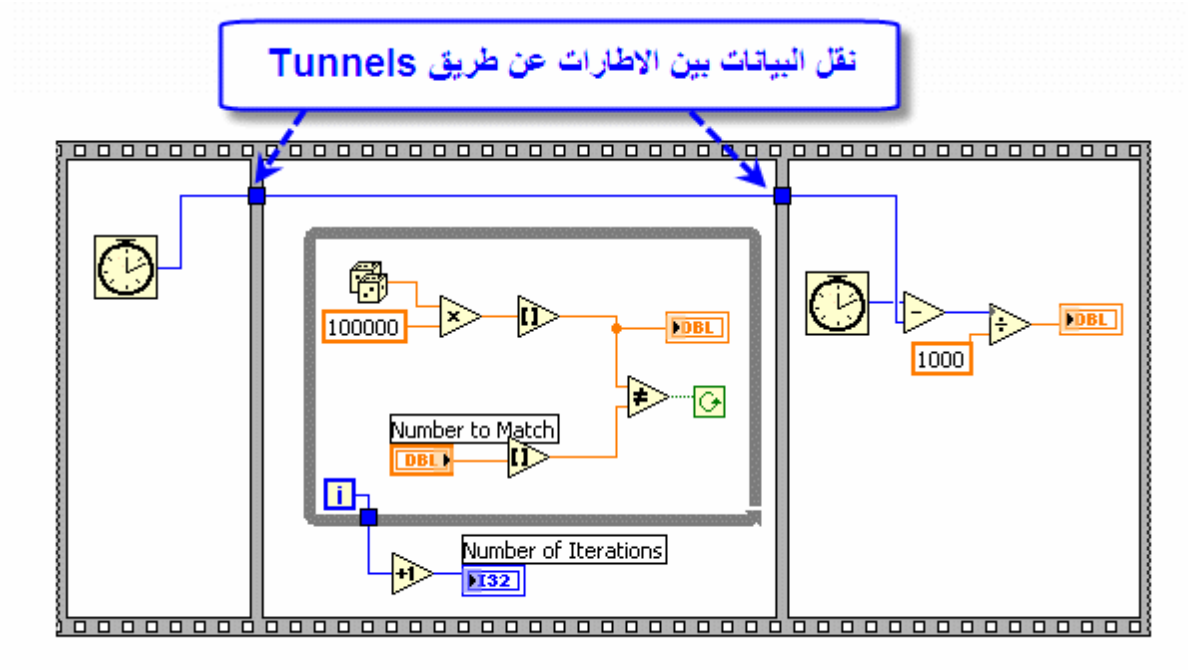
After او Add Frame Before





: Sequence Local

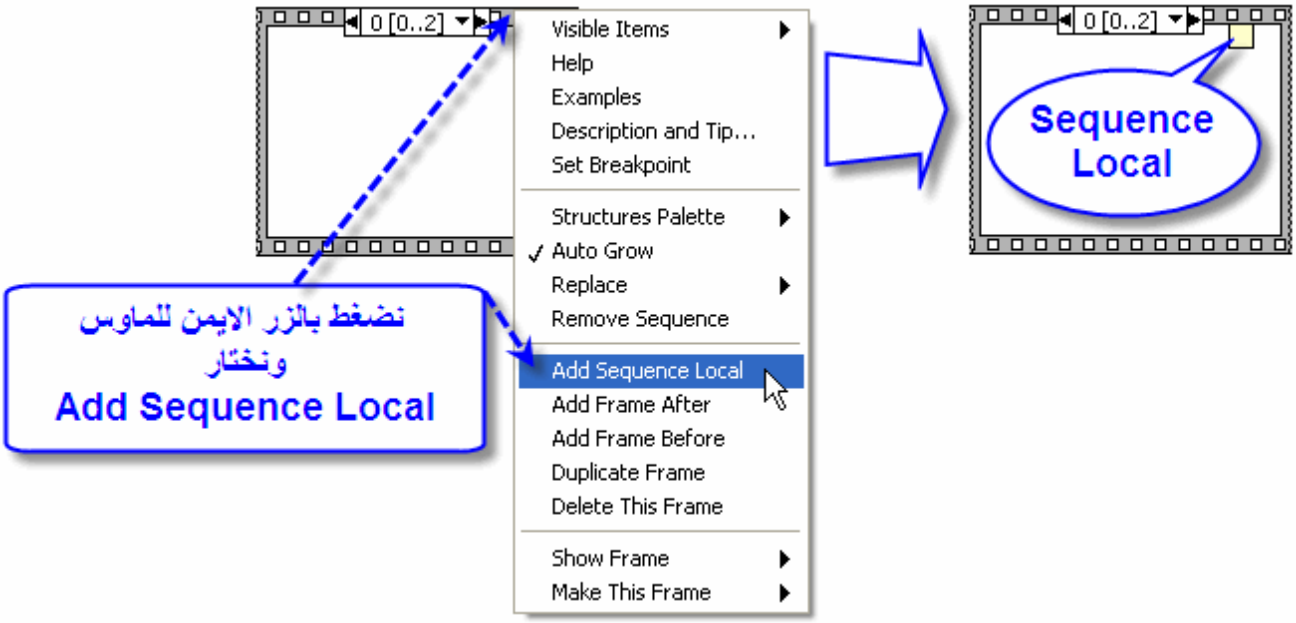
يتم نقل البيانات بين الاطارات فى Flat Sequence Structure بالربط بين الاكواد عن طريق Tunnels (ربط مباشر).



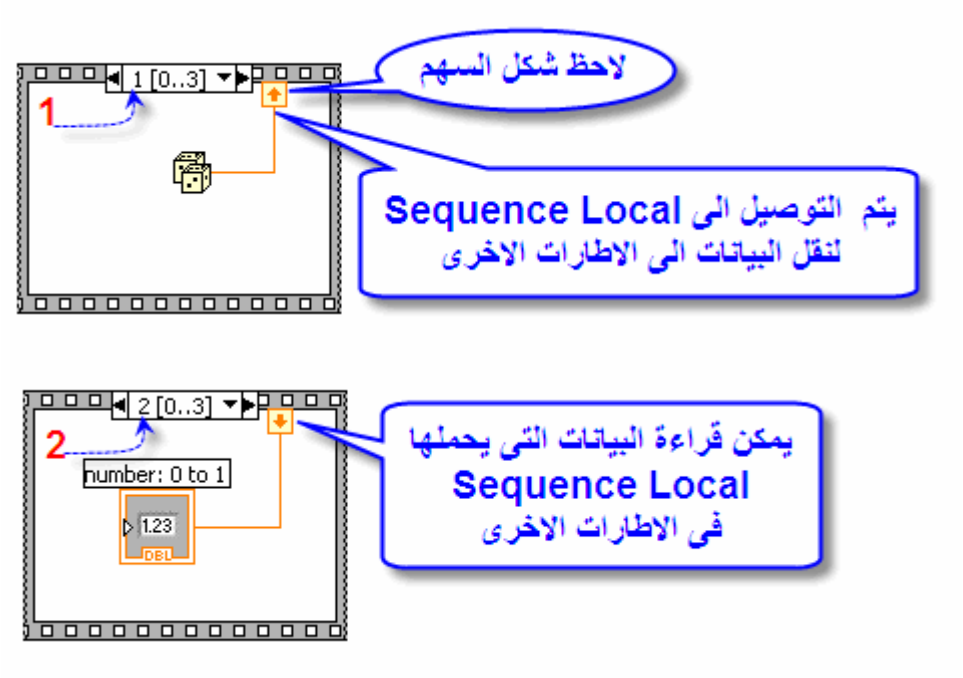
اما فى حالة Stacked Sequence Structure فيستخدم Sequence Local لنقل البيانات بين الاطارات المختلفة.

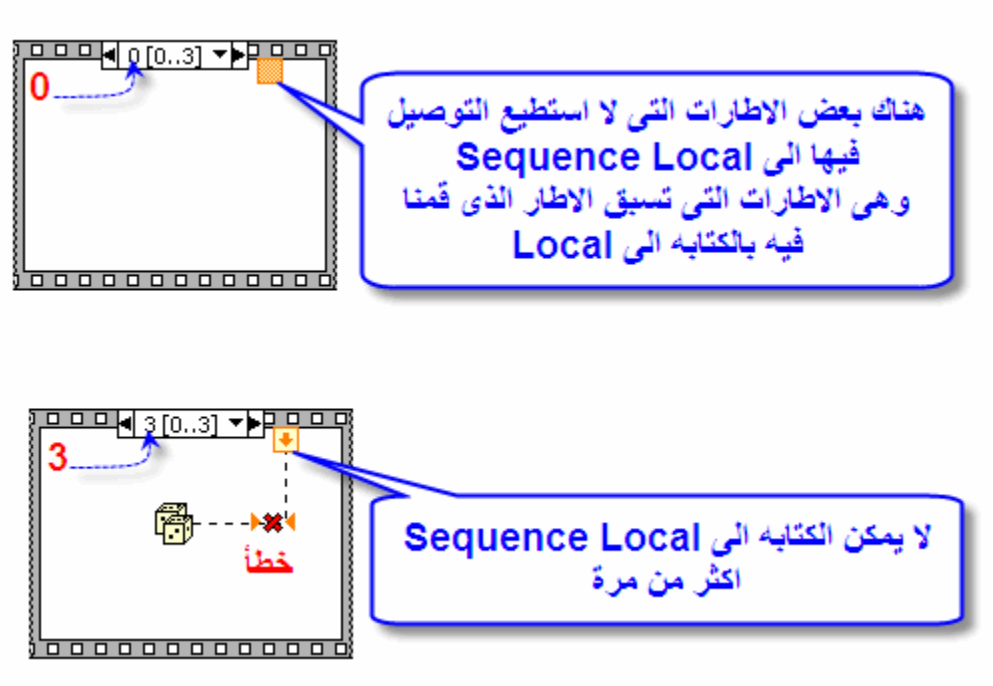
اضافة Sequence Local :

نضغط بالزر الايمن للماوس على الاطار ونختار Add Sequence Local



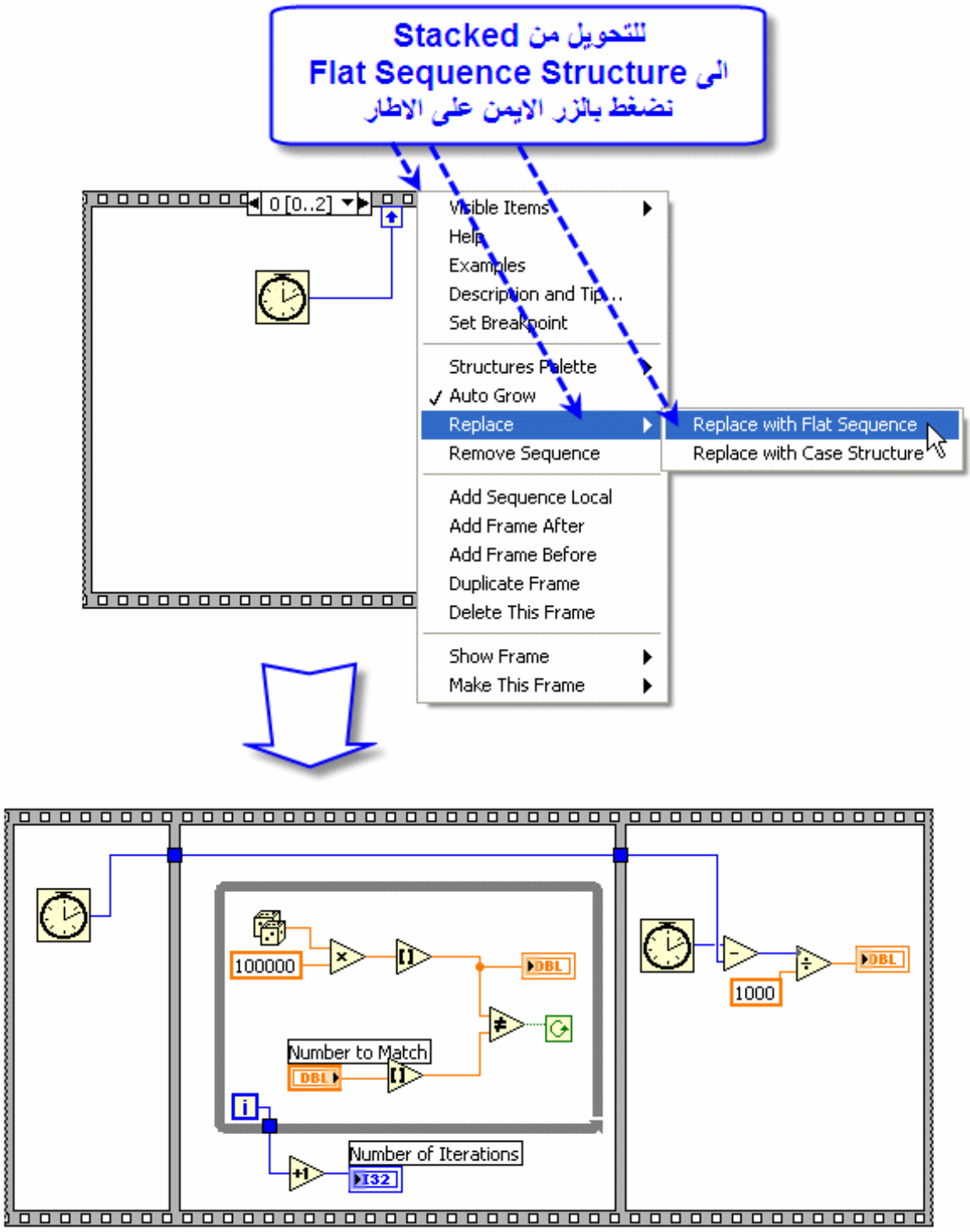
نوصل قيمة الى Sequence Local فى احد الاطارت ويتم قراءة هذة القيمة فى الاطارات الاخرى.





ملحوظة :

فى اى وقت اثناء تصميم البرنامج يمكن التحويل بين Flat and Stacked Sequence Structures.


للتحويل من Stacked
الى Flat Sequence Structure
تضغط بالزر الايمن على الاطار
0 [0..2]
Visible Items
Help
Examples
Description and Tip...
Set Breakpoint
Structures Palette
Auto Grow
Replace
Remove Sequence
Add Sequence Local
Add Frame After
Add Frame Before
Duplicate Frame
Delete This Frame
Show Frame
Make This Frame
Replace with Flat Sequence
Replace with Case Structure
100000
DBL
Number to Match
DBL
i
Number of Iterations
I32
1000
DBL

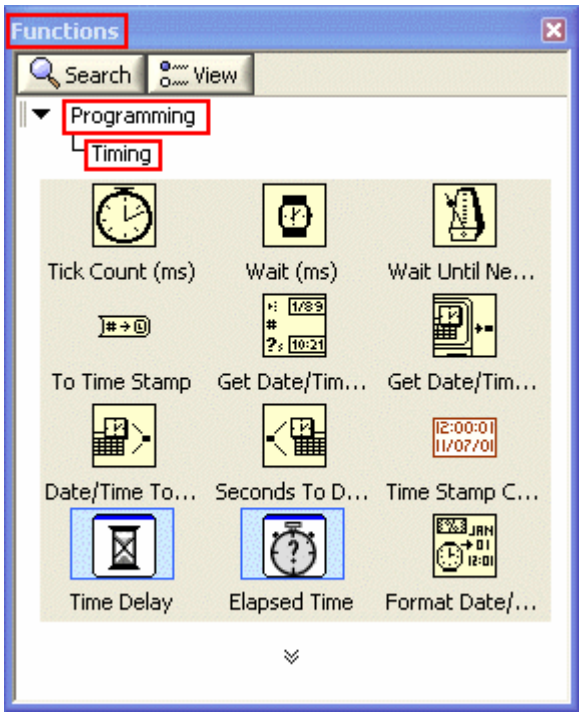
المؤقتات Timing

للمؤقتات اهمية كبيرة فى LabVIEW فهى تستخدم لقياس الزمن وعمل تزامن بين المهام المختلفة و عمل تأخير فى عمل Loops لكى تعمل Loopsبمعدل مناسب حتى لا تستحوز على السرعة الكلية للمعالج.

ويوجد نوعان من المؤقتات : الدوال الاساسية وVIs Express Timing

ويمكن ادراج جميع المؤقتات من

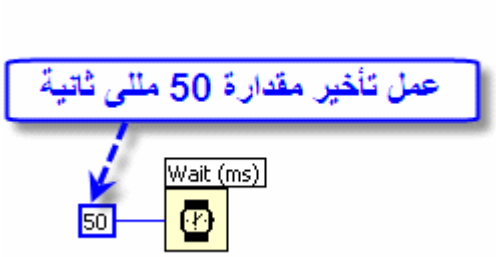
Function Palette >> Programming>> Timing

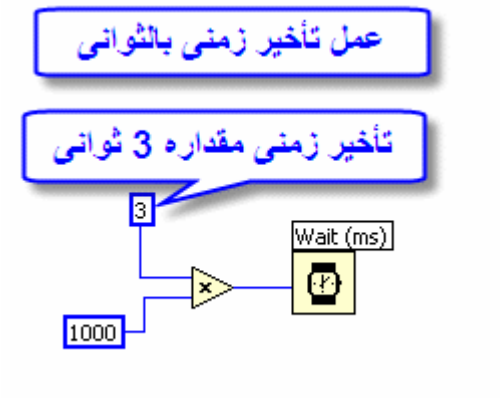


الدوا ل الاساسية :

1. دالة Wait (ms) :

تجعل هذة الدالة VI تنتظر عدد معين من ميللى ثانية ثم بعدها يستمر تنفيذ VI.اى انها تقوم بعمل تأخير زمنى بالمللى ثانية. اذا اردت ان يكون التأخير بالثوان اضرب عدد الثوانى فى 1000.

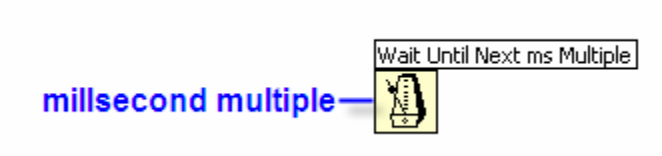




2. دالة Wait Until Next :

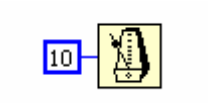
تجعل هذة الدالة VI تنتظر حتى تكون الساعة الداخلية لنظام التشغيل (ويندوز مثلا) تساوى مضاعفات القيمة الداخلة للدالة(millisecond multiple) وبعد ذلك يستمر تنفيذ VI.

وتكون القيمة الداخلة للدالة بالمللى ثانية.



وتتشابة الدالتين Wait(ms) والدالة Wait Until Next ms Multiple ولكن هناك اختلاف بينهما.

ويتضح الاختلاف من المثال التالى:



لو ان القيمة الداخلة لدالة Wait until Next ms Multiple هى 10.

وقيمة ساعة او مؤقت النظام هى مثلا 112.

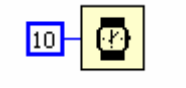
فسوف تنتظر الدالة 8 ميللى ثانية حتى تكون الساعة الداخلية للنظام 120 وهو يعنى مضاعفات 10 (القيمة الداخلة للدالة).

وفى المرة التالية سوف تنتظر الى ان تكون الساعة 130 ايضا مضاعفات 10. وهكذا

لاحظ انه حدث تزامن مع ساعة النظام .

اما بالنسبة للدالة Wait (ms) لو ان القيمة الداخلة لها هى 10.

وقيمة ساعة او مؤقت النظام هى مثلا 112.

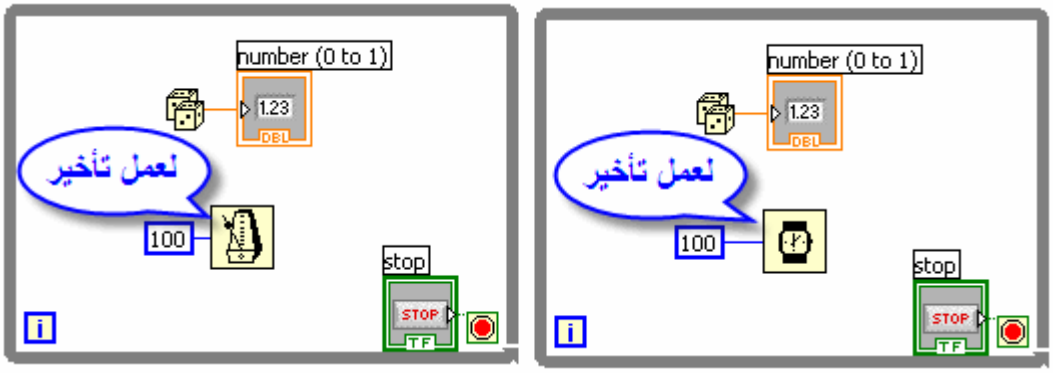


فسوف تنتظر الدالة 10 ميللى ثانية لكى تكون الساعة الداخلية للنظام 122.اى انها انتظرت 10 مللى ثانية بغض النظر عن قيمة ساعة النظام. وهكذا ستنتظر فى كل مرة تنفذ 10 مللى ثانية بغض النظر عن قيمة ساعة النظام.

لذلك تستخدم الدالة فى جعل ما بداخل Loops ينفذ فى ازمنة محددة .كما انها تستخدم فى عمل تزامن بين الاحداث.

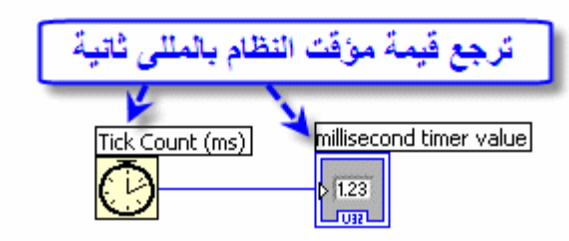
لاحظ انه عندما تستخدم هذة الدالة فى Loop فأنها فى اول مرة تنفذ فيها قد تنتظر اقل من القيمة الداخلة اليها كما فى المثال السابق.

تستخدم الدالتين السابقتين بكثرة فى Loops وذلك حتى لا تستحوز Loop على سرعة المعالج. فتقوم هذة الدوال بعمل التاخير اللازم لعمل المهام الاخرى مثل تفاعل عناصر Front Panel مع المستخدم.



3. دالة Tick Count (ms) :

تستخدم هذة الدالة للحصول على (الساعة) المؤقت الداخلى لنظام التشغيل وذلك بالمللى ثانية.



تستخدم هذة الدالة فى حساب الوقت المنقضى فى تنفيذ جزء معين من البرنامج.

: Express Timing Functions

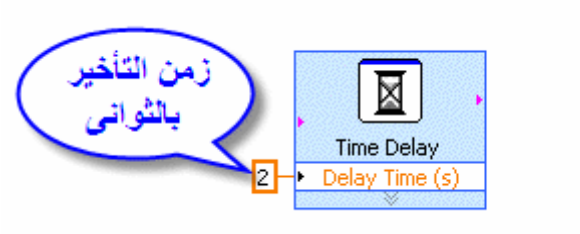
بالاضافة الى الدوال الاساسية الموجودة فى بيئة LabVIEW توجد Express Timing VIs وهما Time Delayو Elapsed Time.

1. Time Delay Express VI :

هى تماثل بالضبط دالة Wait(ms) والفرق بينهما ان فى هذة Express VI نحدد لها زمن التأخير بالثوانى.

وعند ادراج Express VI تظهر نافذة نحدد فيها زمن التأخير الذى نريده.

ويمكن توصيل قيمة التأخير الى Express VI وهذة القيمة سوف تحل محل القيمة التى حددنا بواسطة النافذة السابقة.



2. Elapsed Time Express VI :

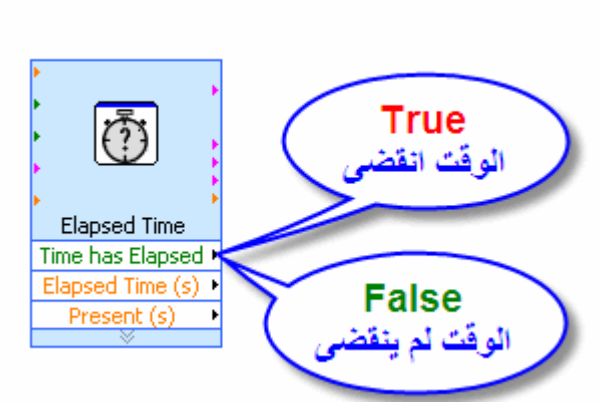
تتيح هذة Express VI معرفة اذا كان قد انقضى وقت (محددة قيمته من قبل) ام لا.

عند ادراج تلك Express VI تظهر لنا نافذة نحدد فيها الوقت الذى نريد معرفة هل انقضى هذا الوقت ام لا.

وتحدد القيمة Time Has Elapsed هلى انقضى الوقت ام لا

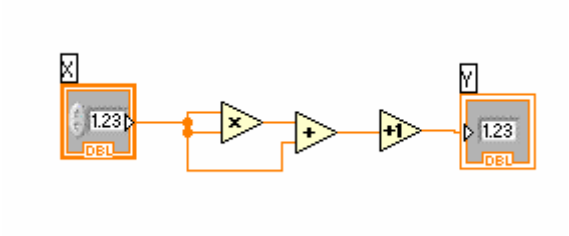
فاذا كانت True فإن الوقت المحدد قد انقضى

اما اذا اكانت False فإن الوقت لم ينقضى بعد.

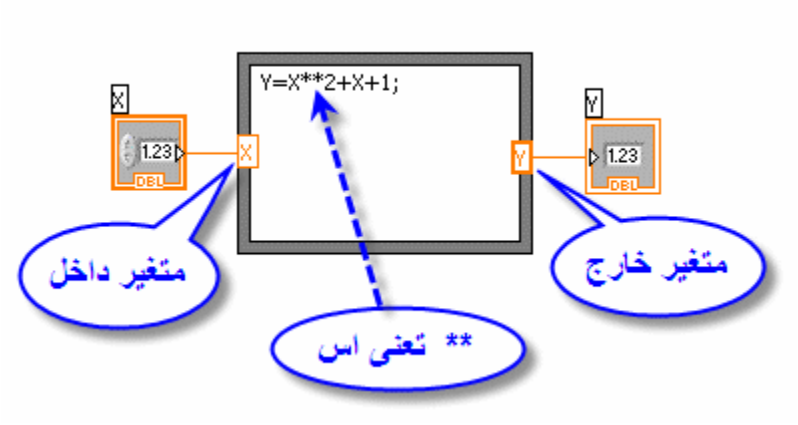


Formula Node

تستخدم Formula Node لكتابة المعادلات الرياضية .
مثلا لتطبيق هذة المعادلة $Y=X^2+X+1$ بواسطة الدوال الاساسية فى LabVIEW فسوف تكون بهذا الشكل



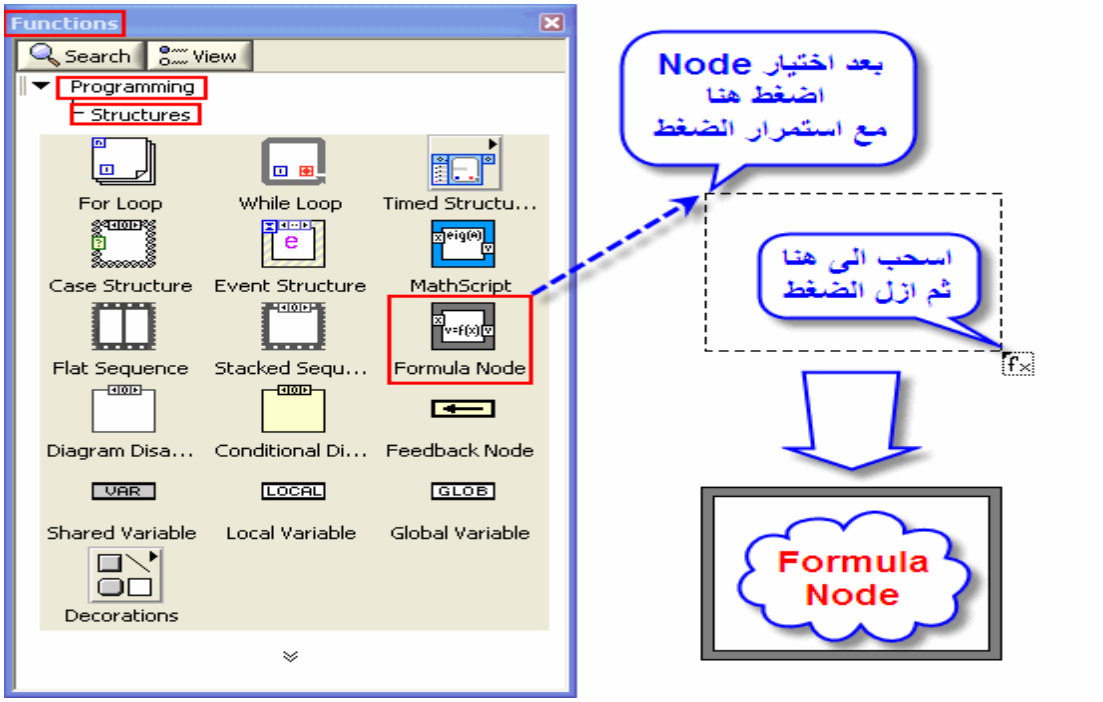
ولكن باستخدام Formula Node ستكون بهذا الشكل



كيفية ادراج Formula Node :

يتم ادرج Formula Node من

Factions Palette >>Programming >>Structures

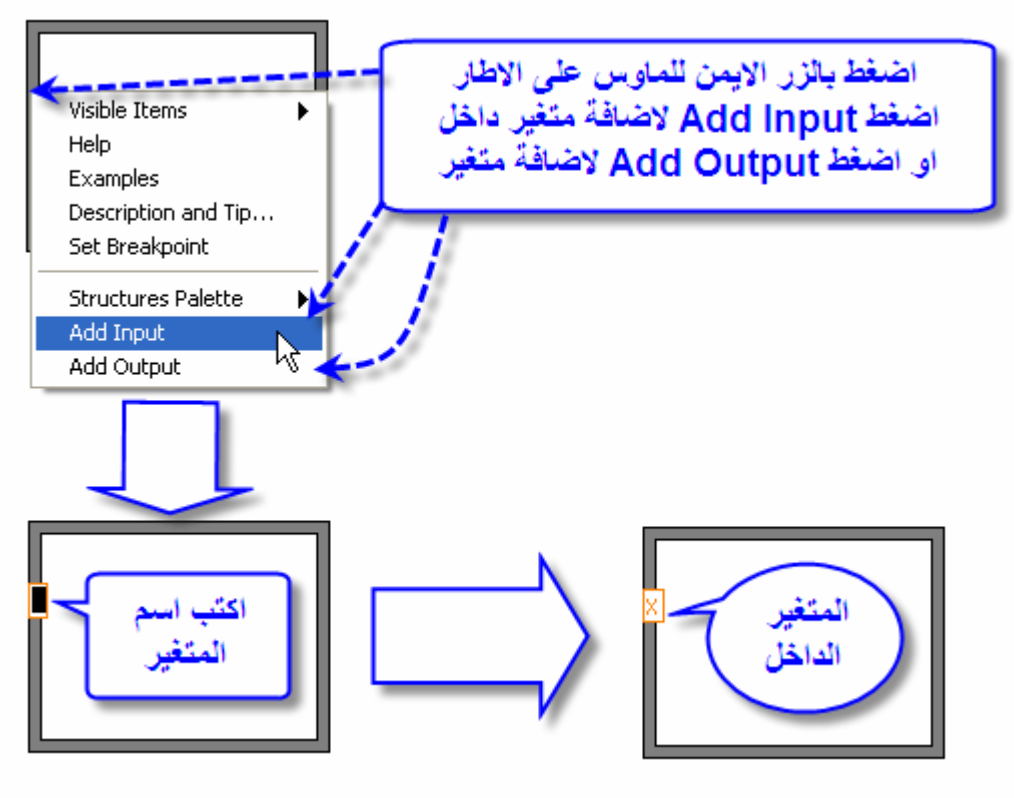


اضافة متغيرات داخلة ومتغيرات خارجة:

يمكن اضافة اى عدد من المتغيرات الداخلة اوالخارجة وذلك بالضغط بالزر الايمن للماوس واختيار

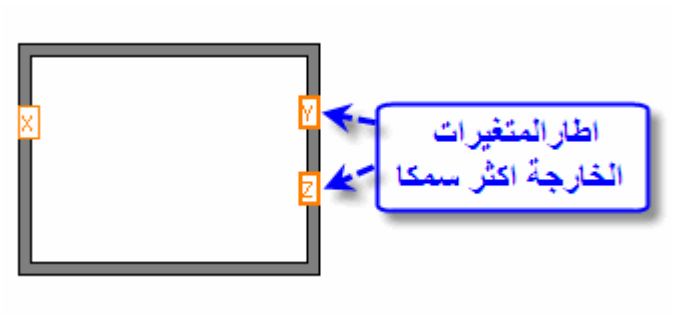
Add Input لاضافة متغير داخل

Add Output لاضافة متغير خارج

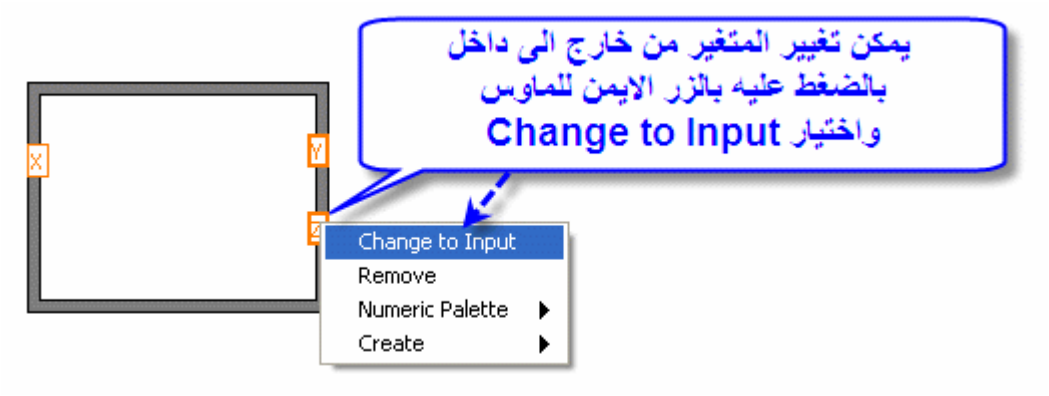


ملحوظات

- لا يمكن ان يكون هناك متغيرين داخلين او خارجين بنفس الاسم .
- يمكن ان يكون متغير خارج واخر داخل بنفس الاسم .
- المتغيرات الخارجة يكون اطارها اكثر سمكا من المتغيرات الداخلة .

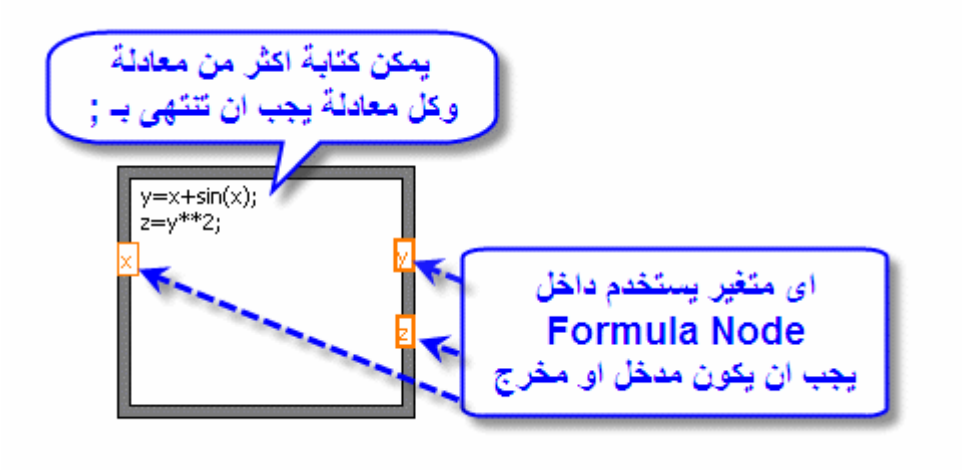


- يمكن تغير المتغير من داخل الى خارج او العكس

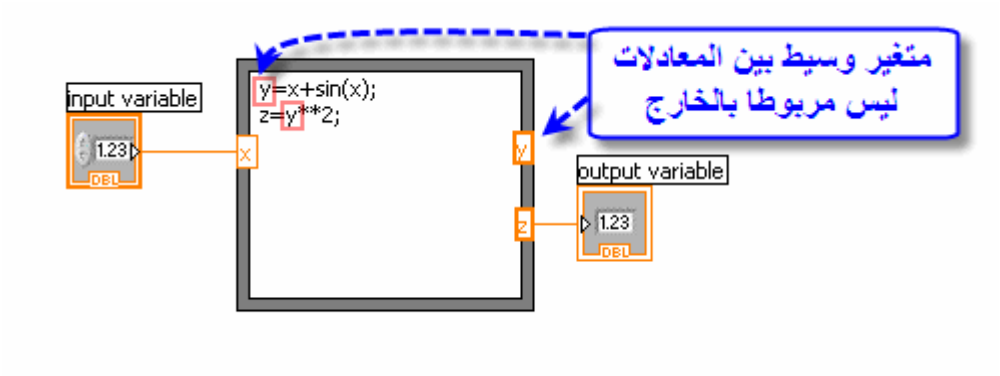


كتابة المعادلات داخل Forumla Node :

تكتب المعادلات فى داخل Fromula Node بطريقة مشابهة لكتابة الجمل فى لغة C.
تكتب المعادلة باستخدام الدوال و اسماء المتغيرات.
يوضع بعد كل معادلة الفاصلة المنقوطة ;
يمكن ان توجد اكثر من من معادلة فى Formula Node.
اى متغير يستخدم فى المعادلات يجب ان يوضع كمتغير خارج او داخل على اطراف Formula Node.



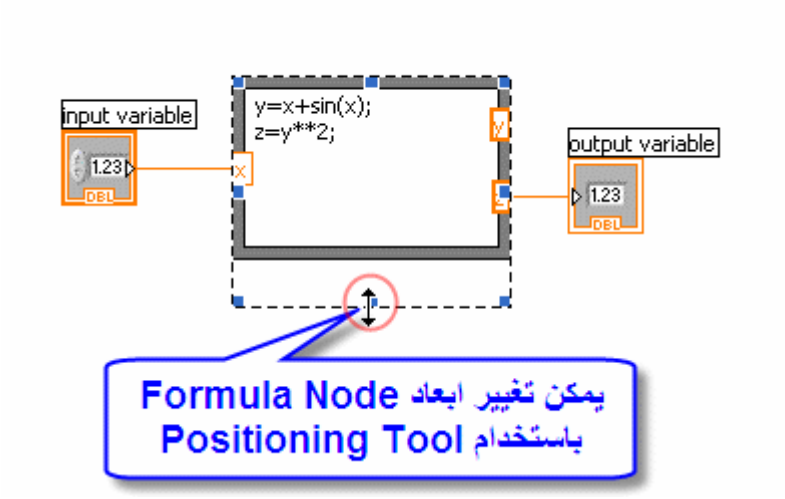
ملحوظة :
ليس حتميا ربط المتغيرات الخارجة بخارج Node فهناك مثلا المتغيرات الوسيطة التى تستخدم بين المعادلات.



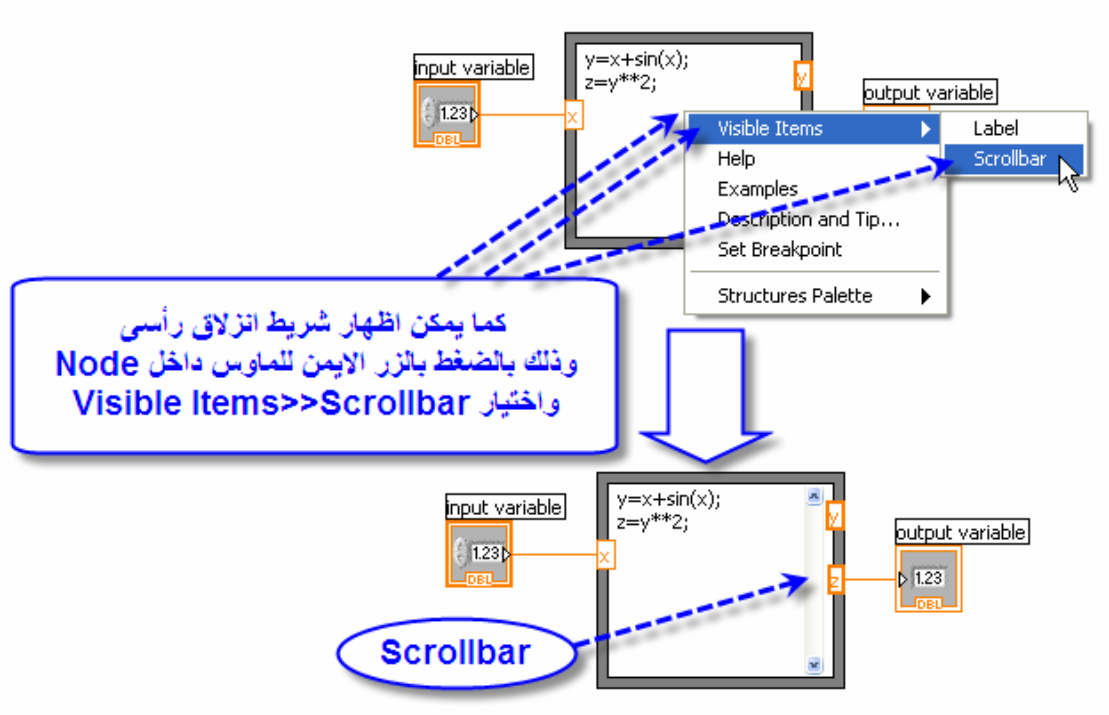
ملحوظة :

ليس هنا عدد محدد لعدد المتغيرات او عدد المعادلات المستخدمة فى Formula Node.

يمكن تكبير Fromula Node لتسع اى عدد من المعادلات وذلك باستخدام Positioning Tool



كما يمكن اظهار شريط انزلاق راسى داخل Node لكتابة اكبر عدد ممكن من المعادلات.



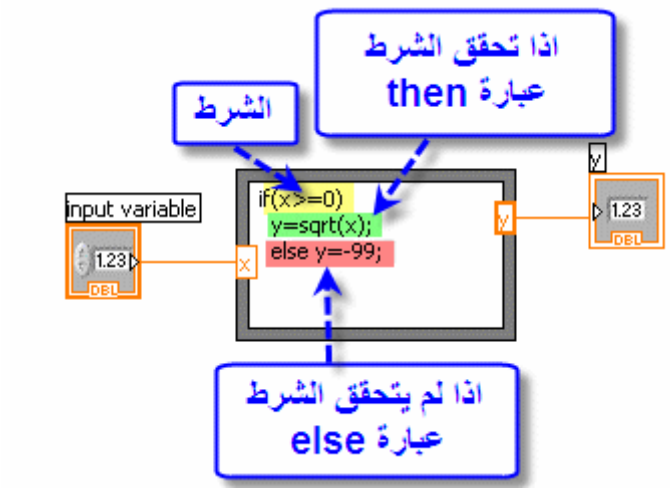
يمكن تطبيق الجملة الشرطية if … then…else داخل Formula Node

فمثلا اذا اردنا تطبيق هذة الجملة

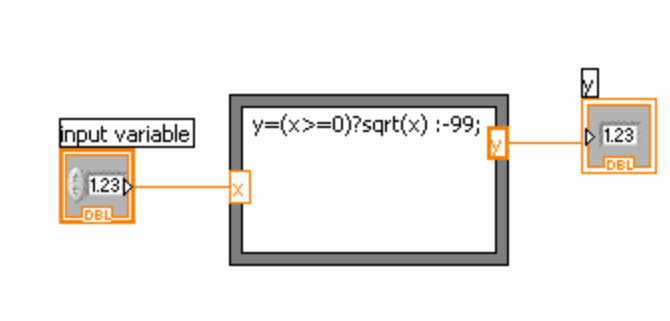
```
If (x>=0) then
        y=sqrt(x)
else
        y=-99
endif
```

والتى تعنى : اذا كانت قيمة المتغير x اكبر من او تساوى 0 فاجعل قيمة المتغير y تساوى الجذر التربيعى لقيمة المتغير x .
واذا كانت قيمة المتغير x غير ذلك اى ليست اكبر ولا تساوى 0 (اى اصغر من 0) فاجعل قيمة المتغير y تساوى الرقم 99- (دلالة على ان المتغير x ليس له جذر تربيعى) .

وهذ المثال يمكن تطبيقه باستخدام Formula Node بهذا الشكل

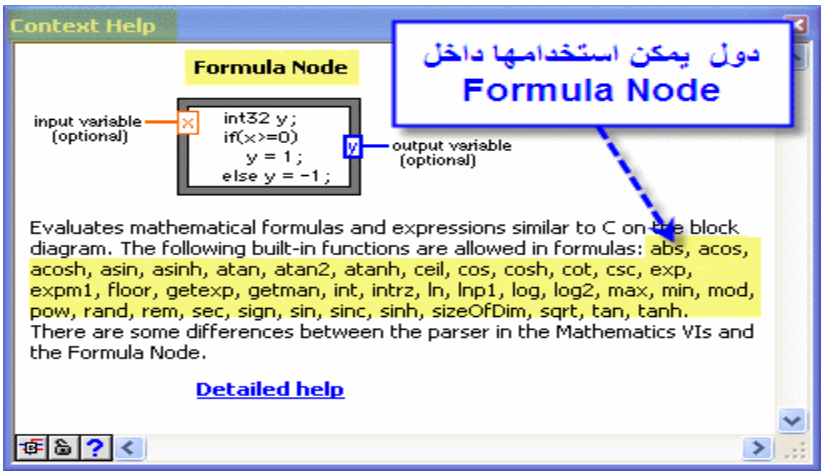


اوبهذا الشكل



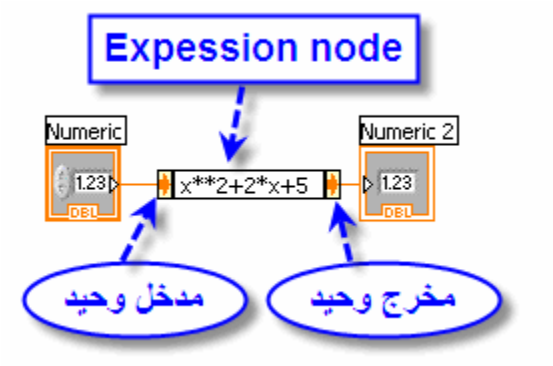
وتتكون الجملة y=(x>=0) ?sqrt(x) : -99 من
الشرط وهو ما بين القوسين X>=0
عبارة Then وهى ما بعد العلامة ? و هى فى هذا المثال قيمة y اذا تحقق الشرط (Sqrt(x)).
عبارة else وهى ما بعد العلامة : وهو فى هذا المثال قيمة y اذا لم يحقق الشرط (-99).

ولمعرفة الدوال التى يمكن ان نستخدمها فى Formula Node نشاهد Context Help لـ Formula Node.



Expression Node

هى صورة مبسطة من Formula Node.فهى تحتوى على عبارة رياضية واحدة ومدخل واحد ومخرج واحد.



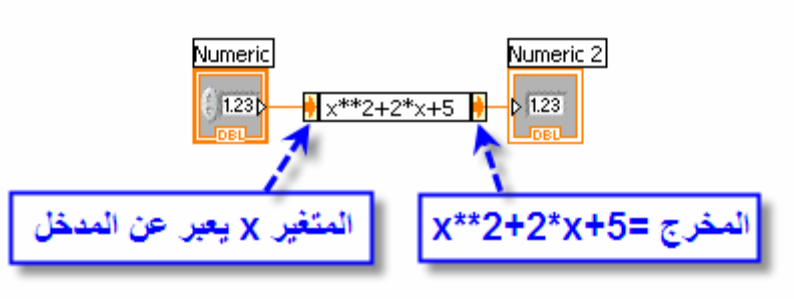
وحيث ان المدخل وان المخرج وحيد فليس لهما اسماء.

وتكتب العبارة الرياضية بدون العلامة ;

ويستخدم متغير وحيد فى هذة العبارة الرياضية ويمكن ان نسمى هذة المتغير باى اسم .

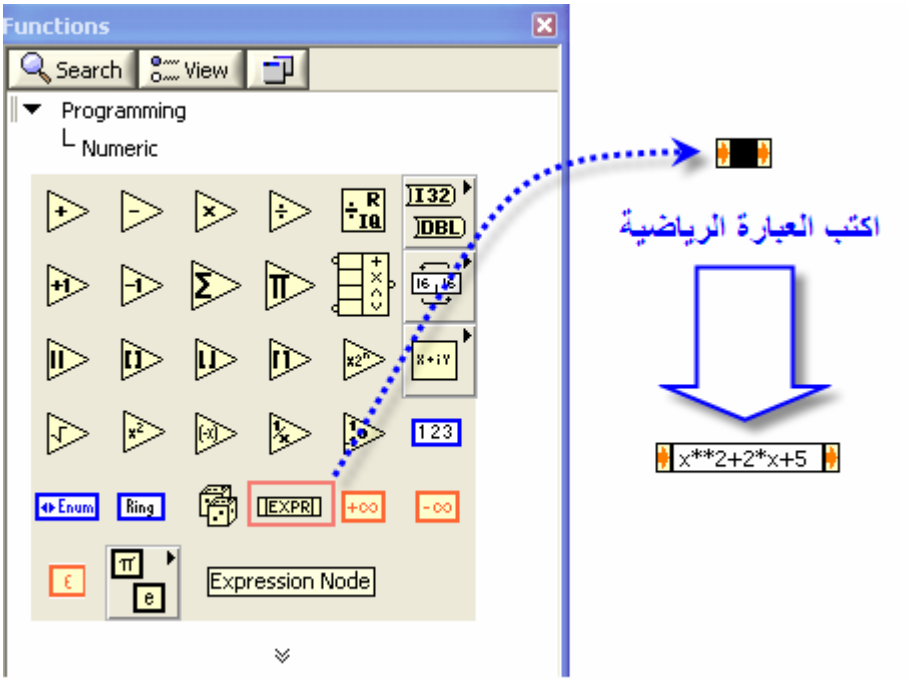
وهذا المتغير يعبر عن المدخل .

والمخرج هو ناتج العبارة الرياضية



ويتم ادارج Expression Node من

Function Palette>>Programming>>Numeric


Functions
Search
View
Programming
Numeric
I32
DBL
123
Enum
Ring
EXPR
+∞
-∞
Expression Node
اكتب العبارة الرياضية
x**2+2*x+5

تم بحمد الله تعالى الانتهاء من الدرس

الى القاء باذن الله تعالى فى الدرس السادس