L'ENVIRONNEMENT LABVIEW

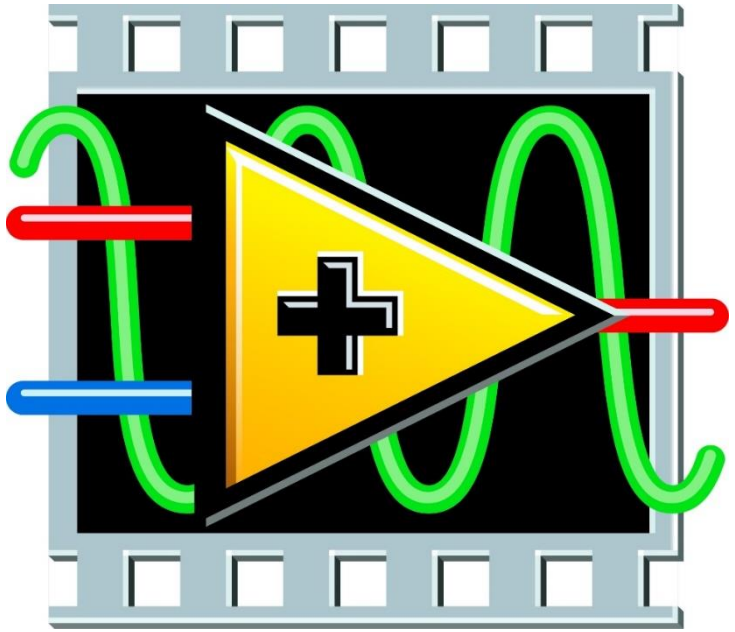
Enseignant : Sébastien Richard

Introduction à LabVIEW

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) / Langage G

Le langage de programmation G est basé sur un environnement de développement graphique développé par la compagnie National Instrument.

NATIONAL INSTRUMENTS LabVIEW



En quoi consiste LabVIEW ?

LabVIEW est un logiciel de développement de systèmes pour les applications de test, de mesure et de contrôle/commande, et permet d'accéder rapidement au matériel et aux informations sur les données.

Il est notamment utilisé pour les systèmes d'acquisition de données.

Contenu de ce tutoriel LabVIEW

1. L'environnement de Labview
2. Les bases de Labview
3. Contrôle de l'éxécution des programmes
4. Structures de données
5. Affichage graphique dans Labview
6. Écriture dans les fichiers
7. Acquisition de données

L'environnement LabVIEW

- Comprendre l'utilisation du *Front Panel* et du *Block Diagram*
- Comprendre la différence entre *contrôles* et *indicateurs*
- Comprendre la notion de programmation *Dataflow*
- Se familiariser avec les menus dans LabVIEW
- Se familiariser avec l'utilisation des différentes palettes
- Réaliser l'importance de l'aide dans LabVIEW

L’environnement LabVIEW

Labview permet de réaliser des VI (pour *Virtual Instrument*), afin de mettre au point le programme que vous désirez concevoir.

Un VI est composé de trois parties liées :

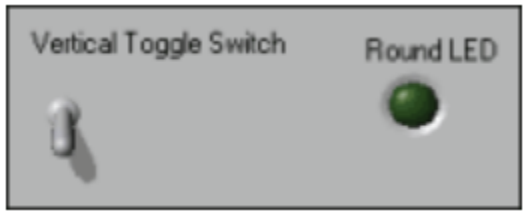
- Une ***face-avant*** (***Front-Panel***) : c'est l'interface utilisateur de la fonction.
- Un ***diagramme*** (***Block-Diagram***) : cette partie décrit le fonctionnement interne du VI.
- Une ***icône*** (***Icon***) : c'est la symbolisation de l'instrument virtuel.

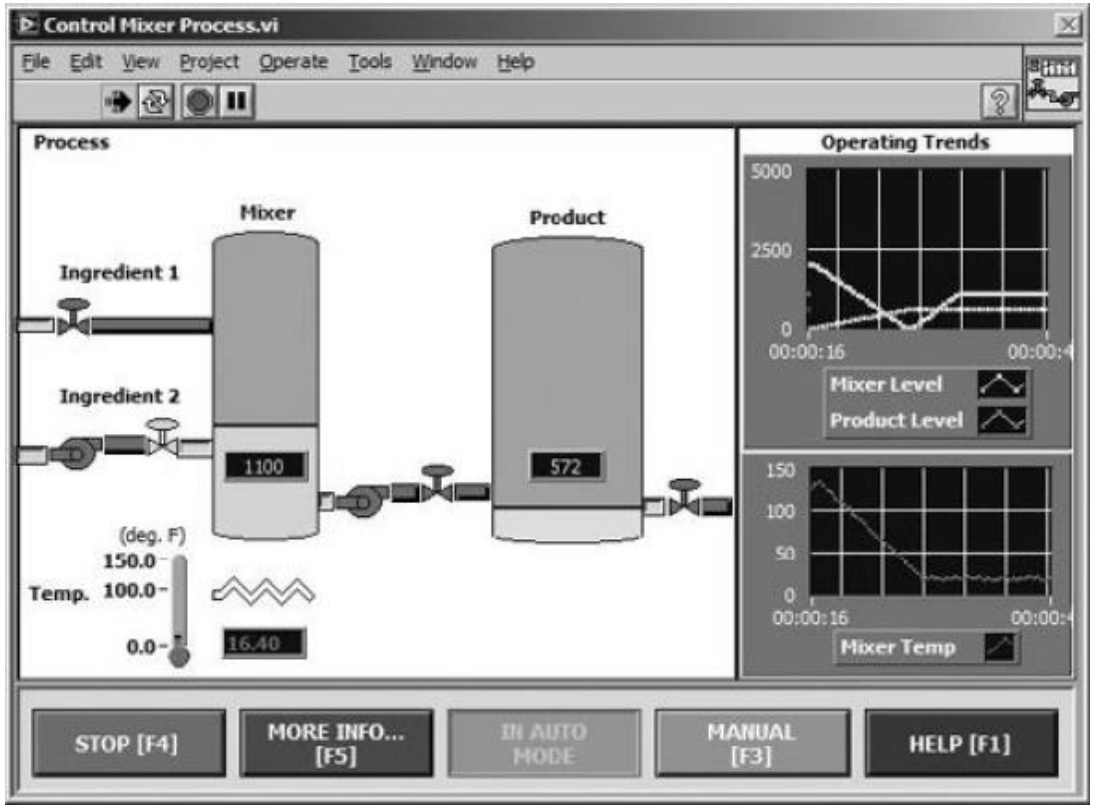
L’environnement LabVIEW – Le Front panel

Le *Front panel* ou face-avant est simplement la « fenêtre » avec laquelle l’utilisateur interagit . Il est très utile car vous pouvez entrer des données à cet endroit mais c’est aussi là où vous pouvez visualiser les résultats de votre programme.

Cette face-avant va nous permettre de mettre au point l'interface utilisateur. Pour ce faire, Labview propose une palette d'outils permettant de manipuler les objets se trouvant sur la face-avant, afin de pouvoir disposer les différentes **commandes** et **indicateurs**, d'éditer le texte ou d'en rajouter et de modifier les couleurs des composants de cette face-avant.

L’environnement LabVIEW – Le Front panel



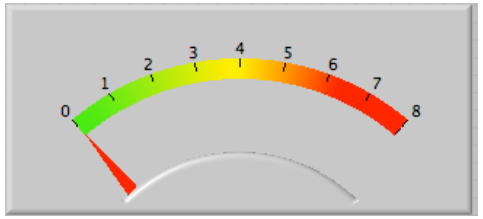


L'environnement LabVIEW – Les contrôles et les indicateurs

Le *Front Panel* est une combinaison de contrôles et d'indicateurs. Vous pouvez modifier les **commandes/contrôles** pour alimenter les entrées et constater le résultat sur les **indicateurs**.

Contrôles = Entrées à partir de l'utilisateur = Source de données

Indicateurs = Sorties vers l'utilisateur = Destinations pour les données



L’environnement LabVIEW – Les contrôles et les indicateurs

Les **contrôles/commandes** définissent les entrées et les **indicateurs** affichent les données en sortie.

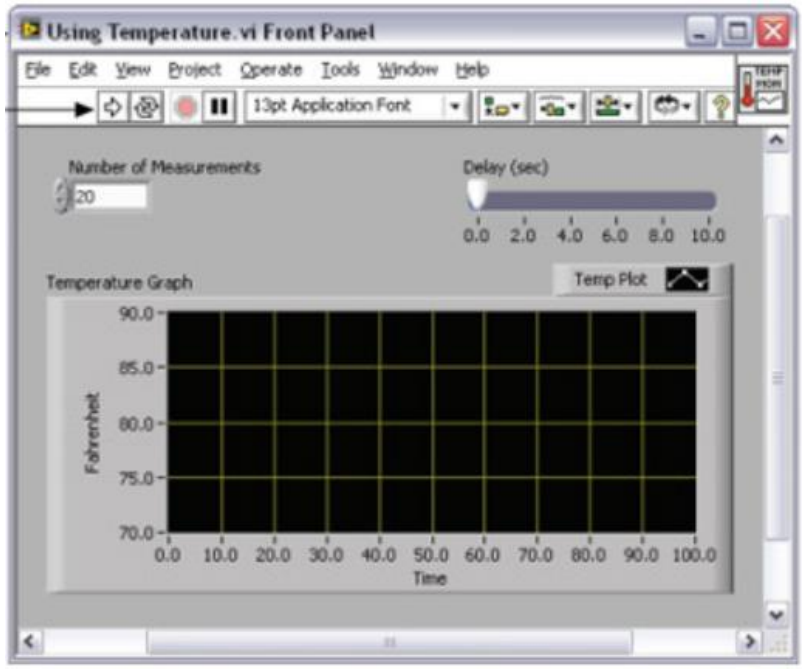
Les **commandes** sont le plus souvent des boutons rotatifs, des boutons poussoirs, des cadrans, des curseurs à glissière et des chaînes de caractères. Elles simulent les éléments d'entrée d'instruments et fournissent des données au diagramme de votre programmation.

Les **indicateurs** sont le plus souvent des graphes, des graphes déroulants, des LED et des chaînes d'état. Les indicateurs simulent les dispositifs de sortie d'instruments et affichent les données que le diagramme acquiert ou génère.

L'environnement LabVIEW – Les contrôles et les indicateurs

La figure ci-dessous, illustre deux commandes : Nombre de mesures et Attente (s). Elle contient un indicateur : un graphe XY nommé Graphe de température.

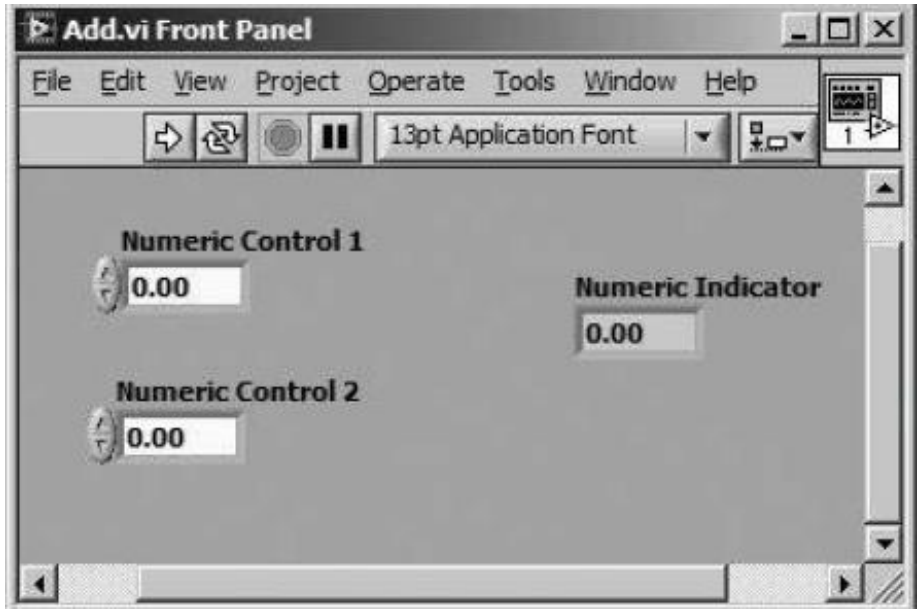
- L'utilisateur peut changer la valeur en entrée des commandes Nombre de mesures et Attente (s). L'utilisateur peut voir la valeur générée par le VI sur l'indicateur Graphe de température. Le VI génère les valeurs des indicateurs en fonction du code créé sur le diagramme.



http://www.ni.com/getting-started/labview-basics/f/environment

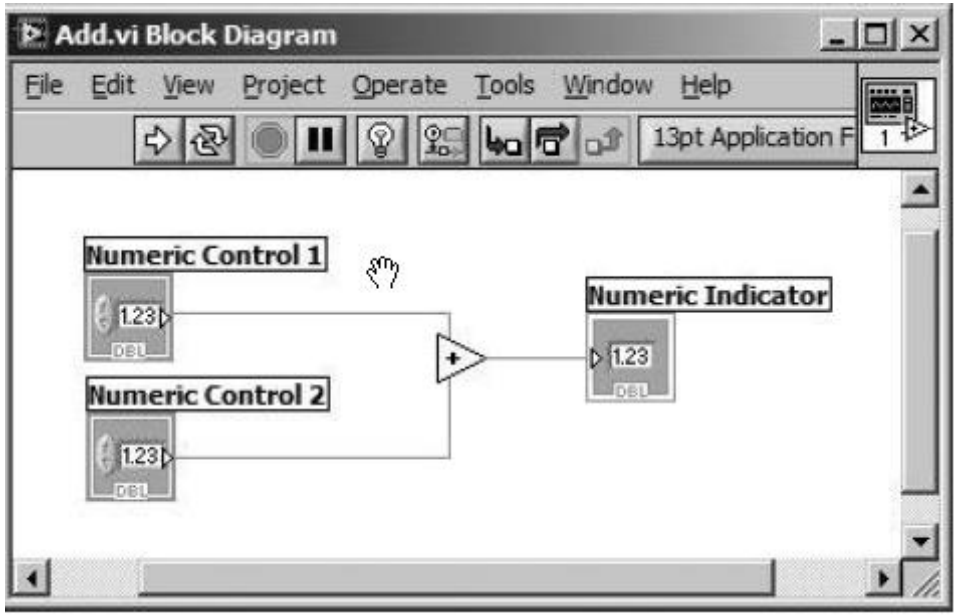
L'environnement LabVIEW – Les contrôles et les indicateurs

Chaque commande ou indicateur a un type de données qui lui est associé. Par exemple, la glissière horizontale Attente (s) est un type de données numérique. Les types de données les plus couramment utilisés sont les types *numérique*, *booléen* et *chaîne*.



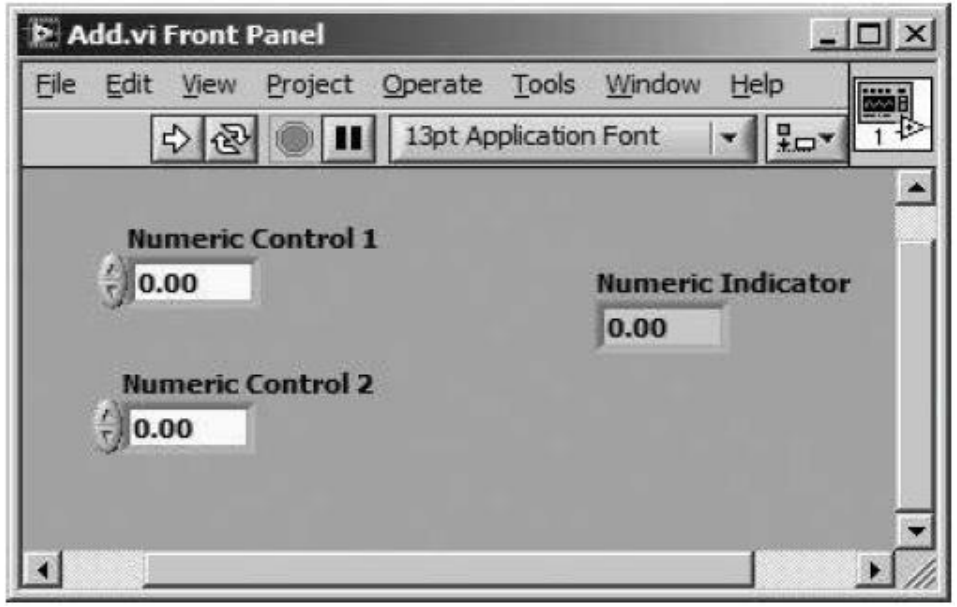
L'environnement LabVIEW – Le Block Diagram

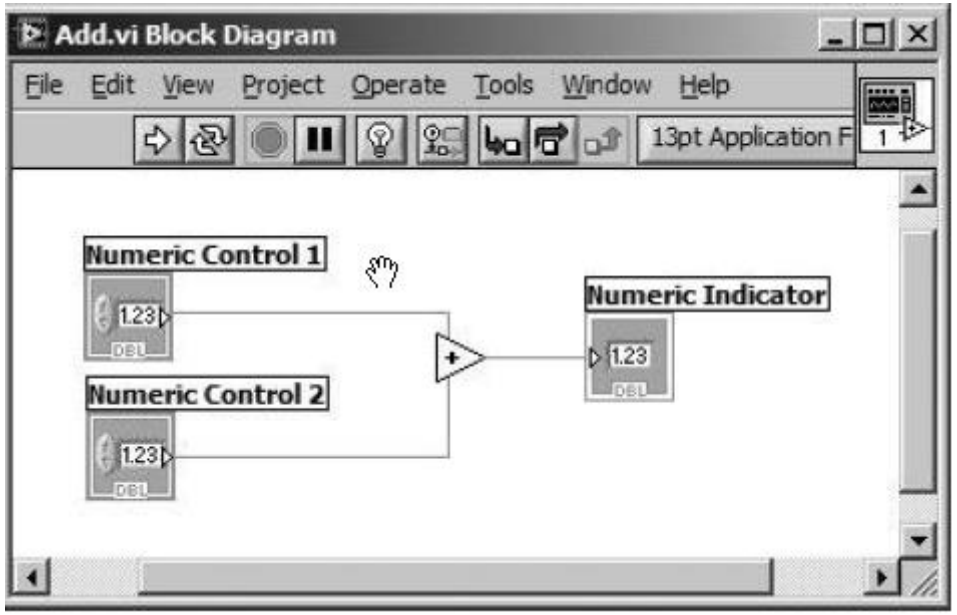
Le *Block Diagram* tient le code source graphique du VI de LabVIEW. Il correspond aux lignes de texte que l'on retrouve dans les logiciels de programmation conventionnels. C'est aussi à cet endroit que vous câblez tous les éléments de votre programme afin qu'ils puissent interagir entre eux.



L'environnement LabVIEW – Block Diagram VS Front panel

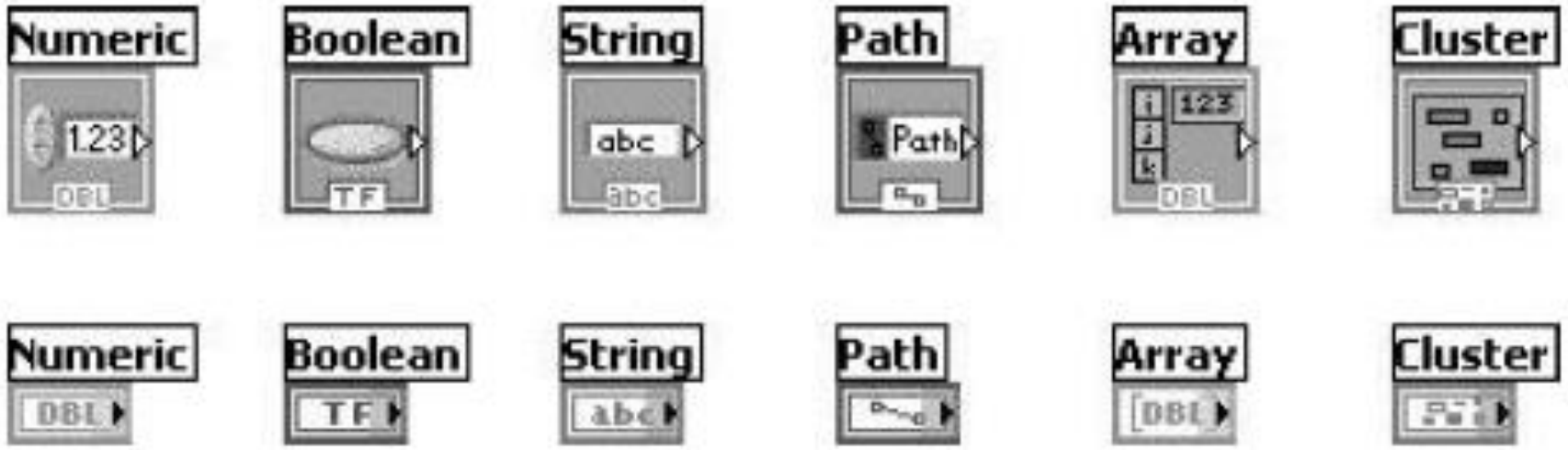
Un bloc diagramme est automatiquement créé dès que l'on crée un front panel. Ces deux éléments sont intimement liés, si l'un est modifié, l'autre le sera aussi.





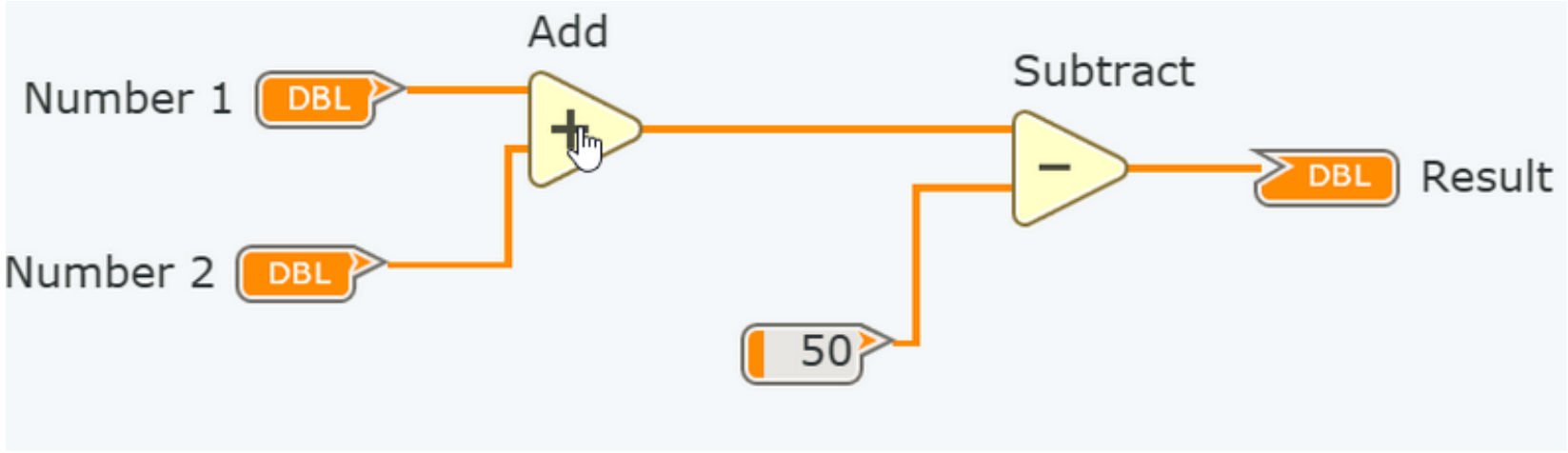
L'environnement LabVIEW – Les terminaux

Vous pouvez voir les *Terminals* comme étant les icônes dans le Block Diagram. Les indicateurs et les contrôle sont associés au Front Panel de la même façon que les Terminals sont associés au Block Diagram.



L'environnement LabVIEW – Les noeuds

Les nœuds sont les éléments d'éxécution d'un programme, tel que les fonctions Add and Substract. Les nœuds sont des objets du diagramme qui comportent des entrées et/ou des sorties et qui effectuent des opérations pendant l'exécution du VI.



L’environnement LabVIEW – Les noeuds

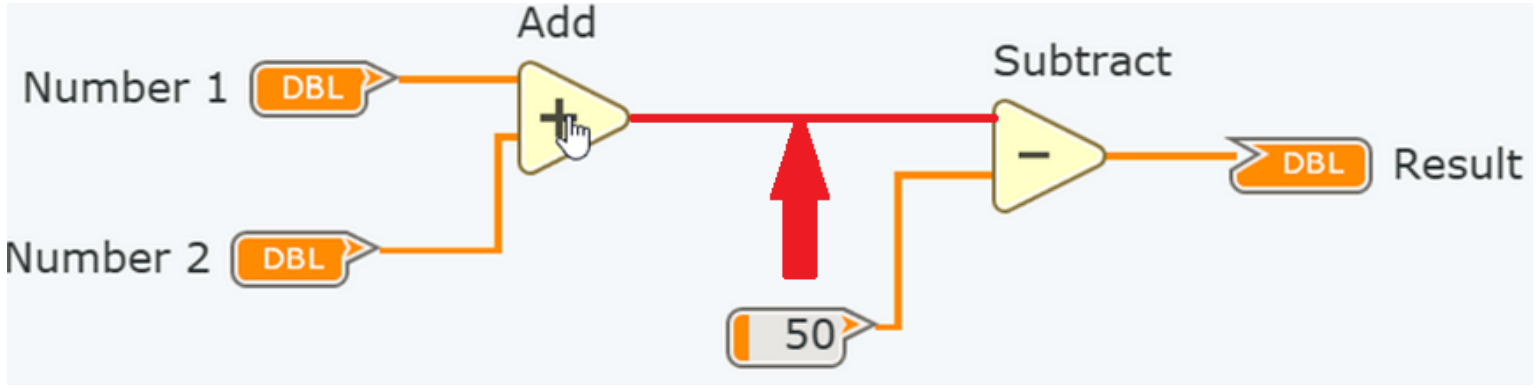
Ils sont semblables aux déclarations, opérateurs, fonctions et sous-programmes écrits dans les langages de programmation textuels. Les nœuds peuvent être des fonctions, des sous-Vis ou des structures.

Les structures sont des éléments de commande du processus, tels que des structures Condition, des boucles For ou des boucles While.

L'environnement LabVIEW – Les fils

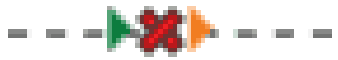
Les wires (fils) sont tout simplement les connexions entre les différents noeuds et les terminaux.

C'est le chemin pour les données entre la source et la destination.



L’environnement LabVIEW – Les fils

Un fil de liaison brisé apparaît sous forme de trait noir en pointillés avec un X rouge au milieu, comme représenté ci-dessous. Les fils de liaison peuvent être brisés pour de multiples raisons, notamment lorsque vous essayez de câbler deux objets dont les types de données sont incompatibles.



L'environnement LabVIEW – Les fils

Chaque fil de liaison a une source de données unique, mais vous pouvez le câbler à de nombreux VIs et fonctions capables de lire ces données. Les fils de liaison sont de couleurs, styles et épaisseurs différents en fonction du type de données transféré.

Type de fil de liaison	Scalaire	Tableau 1D	Tableau 2D	Couleur
Numérique				Orange (virgule flottante), Bleu (entier)
Booléen				Vert
Chaîne				Rose

Tableau n°1. Fils de liaison les plus courants

L’environnement LabVIEW – Programmation Dataflow

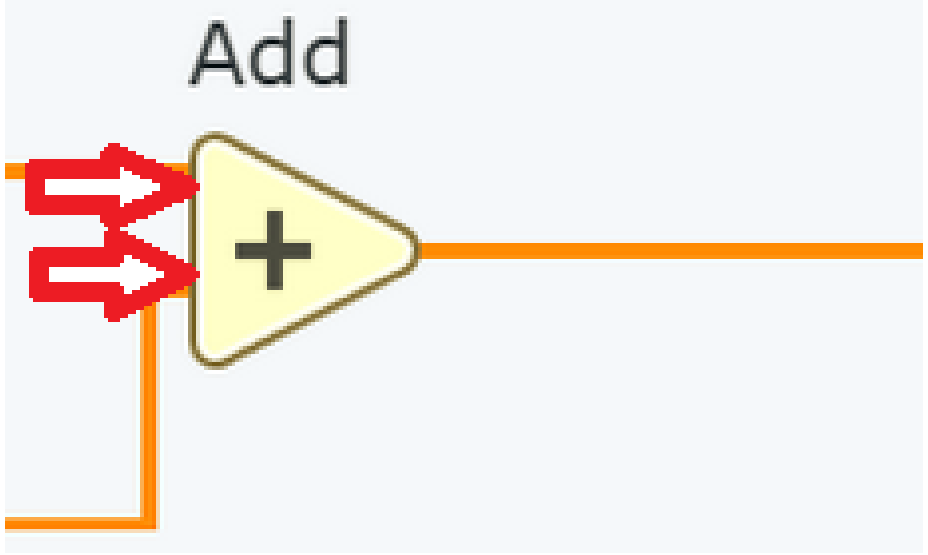
Parce que Labview n’est pas un language de programmation basé-texte, son code n’est pas éxécuté « ligne par ligne ».

Il fonctionne plutôt sur un principe d’éxécution de programme appelé ***Dataflow***.

L’environnement LabVIEW – Programmation Dataflow

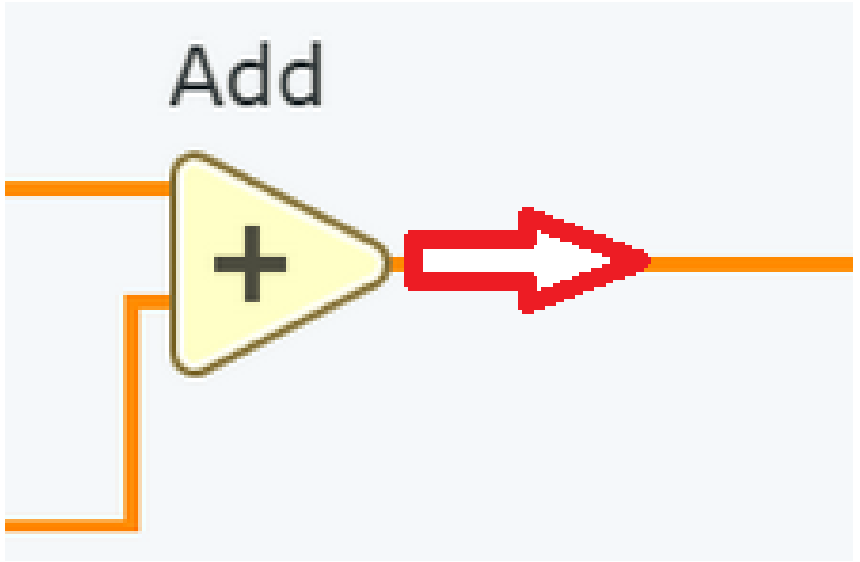
Cela veut dire tout simplement qu’un nœud

s’exécute seulement quand les données arrivent à toutes ses entrées .



L'environnement LabVIEW – Programmation Dataflow

Lorsque les données sont présentes à toutes ses entrées, il exécute sa fonction et fournit les données à toutes ses sorties.

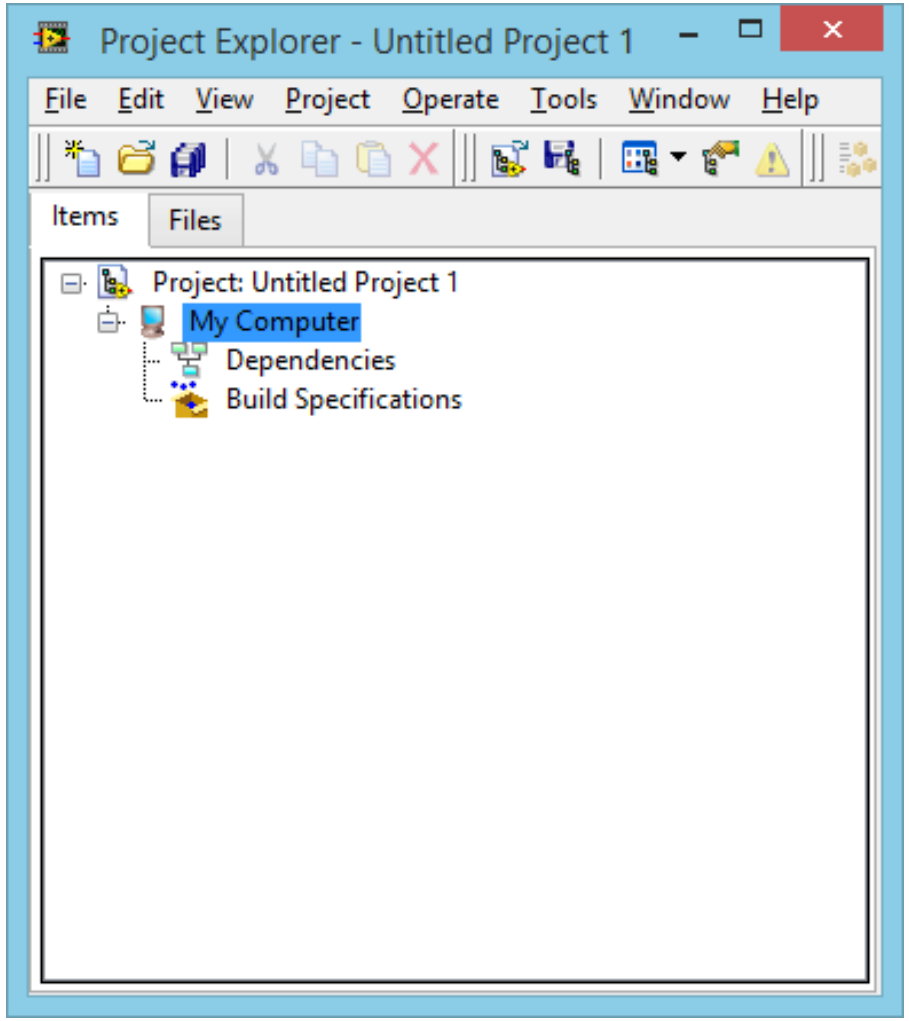


L'environnement LabVIEW – Programmation Dataflow

Bref, on dit qu'il est dépendant de données.

L'environnement LabVIEW – Projet

Lorsque vous créer plusieurs VI(s) pour un seul projet, alors il devient important de les classer correctement dans un projet dans l'Explorateur de projet. Si l'on veut créer un programme avec un exécutable, il est aussi nécessaire de créer un projet. On peut aussi insérer d'autres fichiers utiles tels que : des pilotes, des fichiers d'aide, etc…



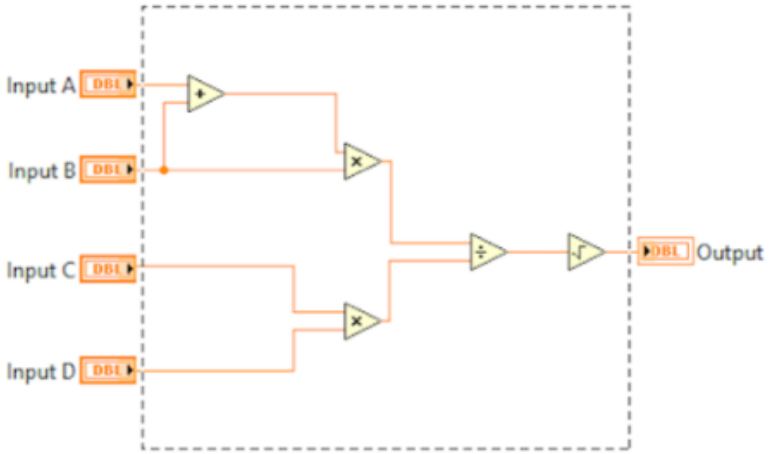
L'environnement LabVIEW – Projet

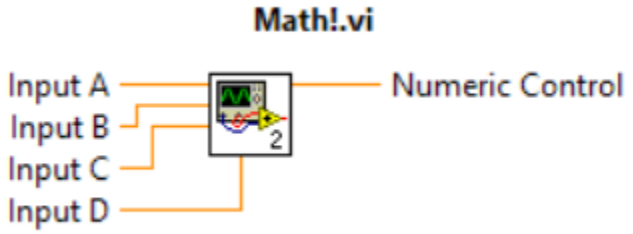
Création de l'arborescence

L'environnement LabVIEW – SubVI

Un SubVI réfère simplement à un VI qui est appelé à interagir avec un autre VI.

Lorsque le VI opère comme étant un SubVI, ses contrôles et ses indicateurs reçoivent et retournent des données au VI qui lui demande.



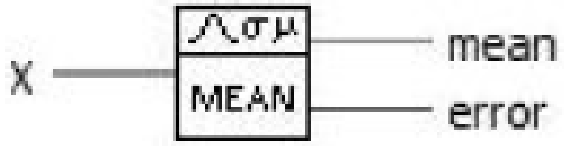


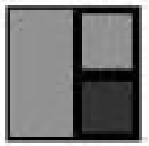
L'environnement LabVIEW – SubVI

Un SubVi vient automatiquement avec une icône, car il devient comme un Terminal dans le Block Diagram.

L'environnement LabVIEW – SubVI/Connector

Comme un terminal standard, le SubVi doit avoir des connecteurs E/S. Pour cela, il faut le spécifier avec le tableau *Connector Pane*.

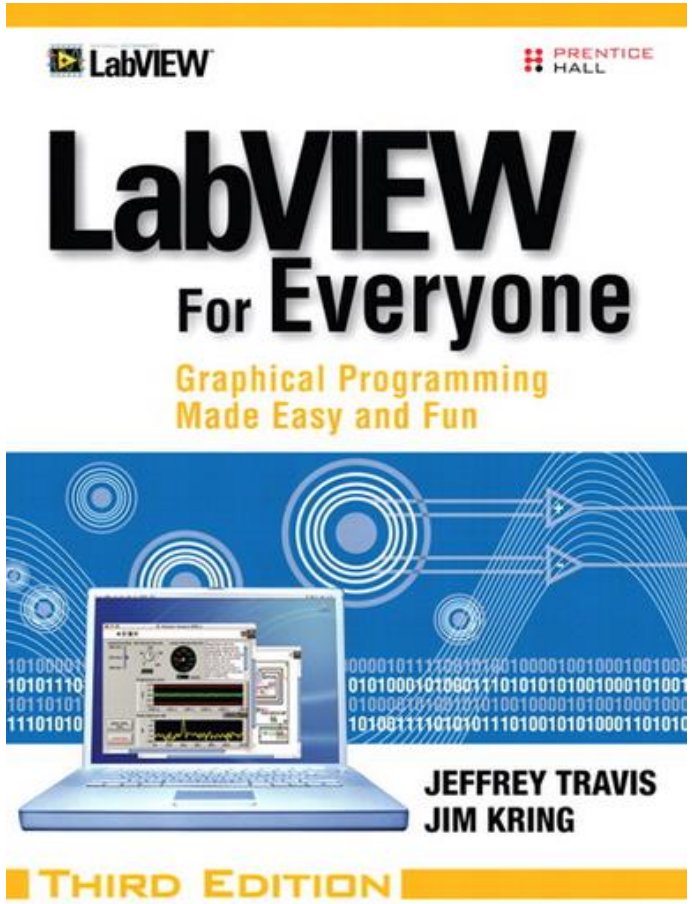




L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

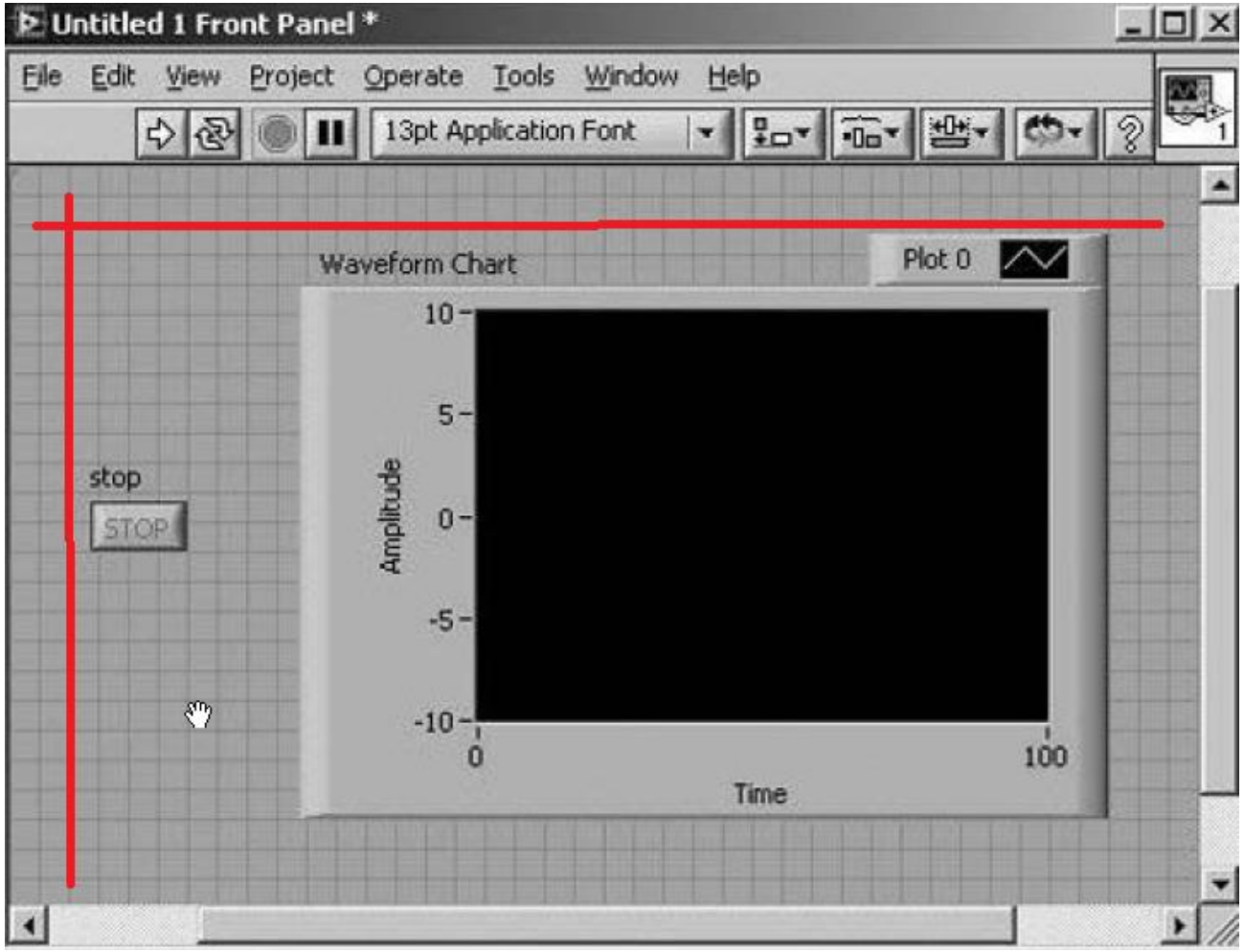
Veuillez vous rendre à la page 54 de votre livre *LabVIEW for Everyone* et compléter l'activité 3-1.



L'environnement LabVIEW – Alignment Grid

La grille d'alignement peut être affichée ou masqué, elle est cependant très utile pour l'arrangement des différents composants de votre interface.

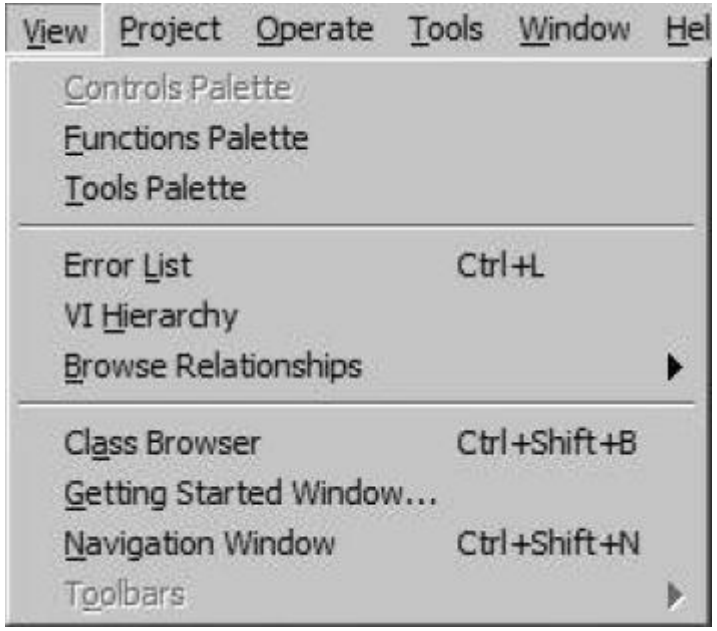
Tools/Options…Front Panel



L'environnement LabVIEW – Pull-Down Menus

Menus déroulants

- File (Save, Print…)
- Edit (Cut, Copy, Paste…)
- View (Controls, Functions et Tools Palette)
- Windows (Fenêtre Front Panel, Block Diagram)
- Help



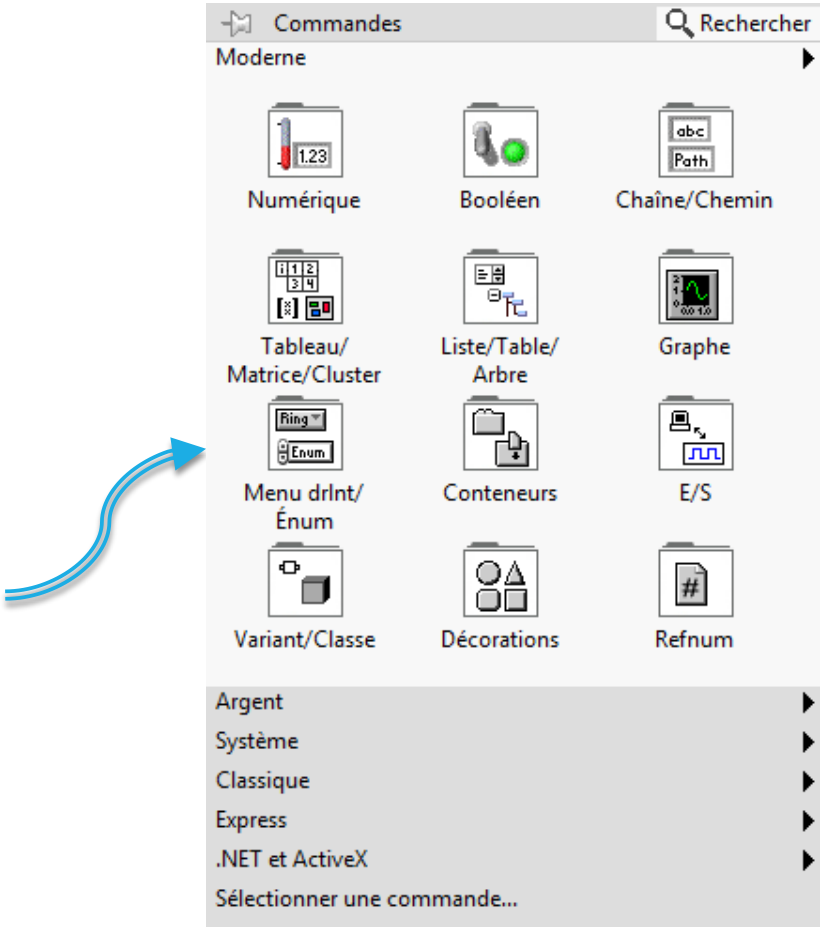
**Veuillez vérifier votre feuille de référence pour les raccourcis.*

L'environnement LabVIEW – Floating Palettes

Controls Palette

Elle est seulement visible dans l'environnement ***Front Panel .***

*Vous pouvez toujours la voir ou la faire disparaître après utilisation.

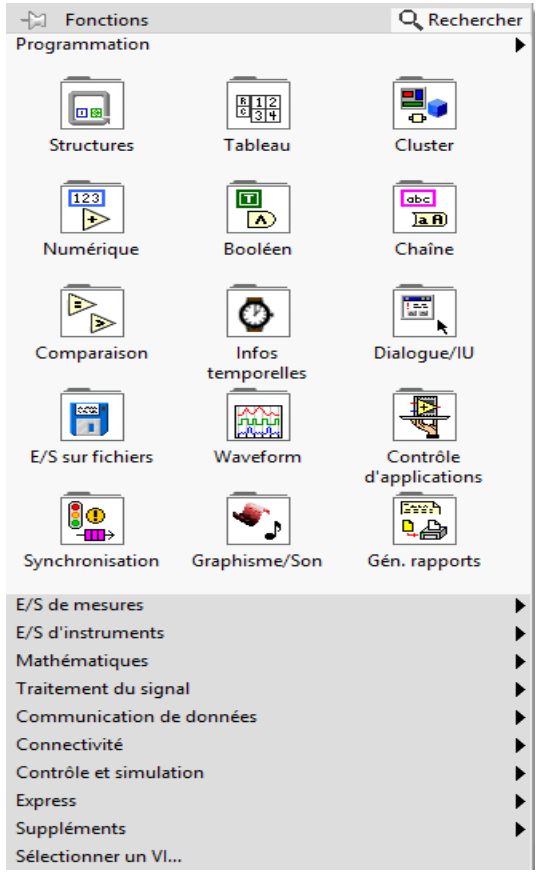


L'environnement LabVIEW – Floating Palettes

Function Palette

Elle est seulement visible dans l'environnement ***Block Diagram.***

*Vous pouvez toujours la voir ou la faire disparaître après utilisation.



L'environnement LabVIEW – Floating Palettes

Tools Palette

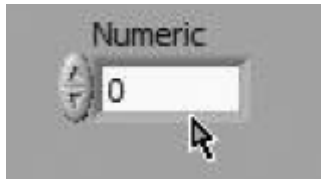
Elle est utilisée dans les deux environnements, soit le ***Front Panel et Block Diagram.***

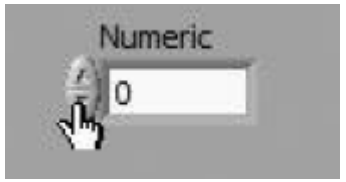
**Liste des outils p.79*

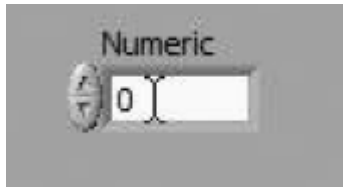
L'environnement LabVIEW – Tools Palettes

Automatic Tool Selection

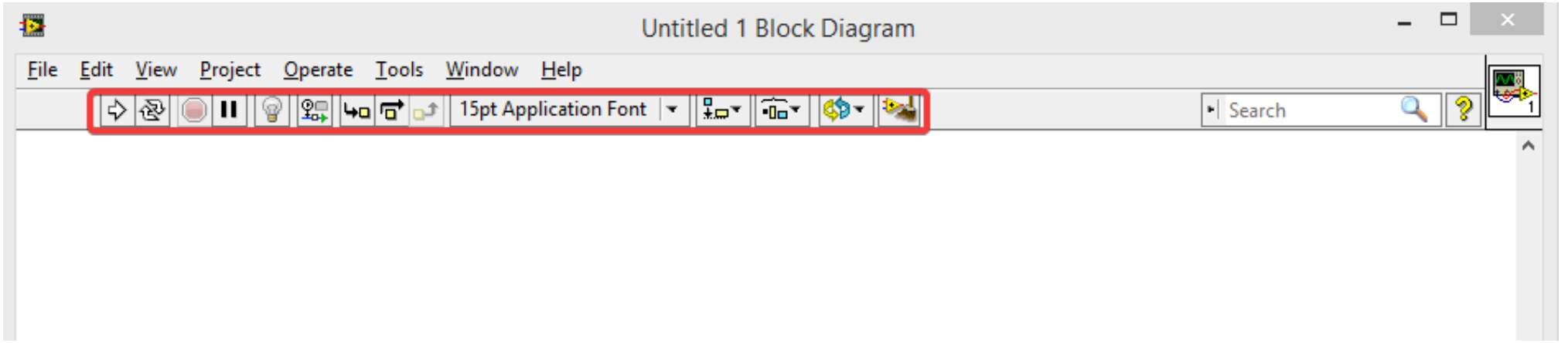
Cette fonction permet que le logiciel sélectionne automatiquement l'outil que vous semblez avoir besoin lorsque vous déplacez votre curseur de souris par-dessus les éléments de votre VI.







L'environnement LabVIEW – Toolbar



Cette barre d'outils, située normalement en-haut à gauche, contient les boutons qui servent à l'exécution de votre programme. La flèche permet de démarrer l'exécution de votre VI.

L'environnement LabVIEW – États du bouton *RUN*

Les 3 états

Run Button

La VI est à l'état repos, elle ne s'exécute pas mais compile.

Run Button (Active)

La VI s'exécute.

Run Button (Broken)

La VI ne peut s'exécuter car elle ne compile pas, vous avez un erreur dans votre programmation

L’environnement LabVIEW

Abort Button 

Sert à arrêter l’exécution de votre programme.

Pause Button 

Sert à mettre en pause l’exécution de votre programme.

Execution Highlight Button 

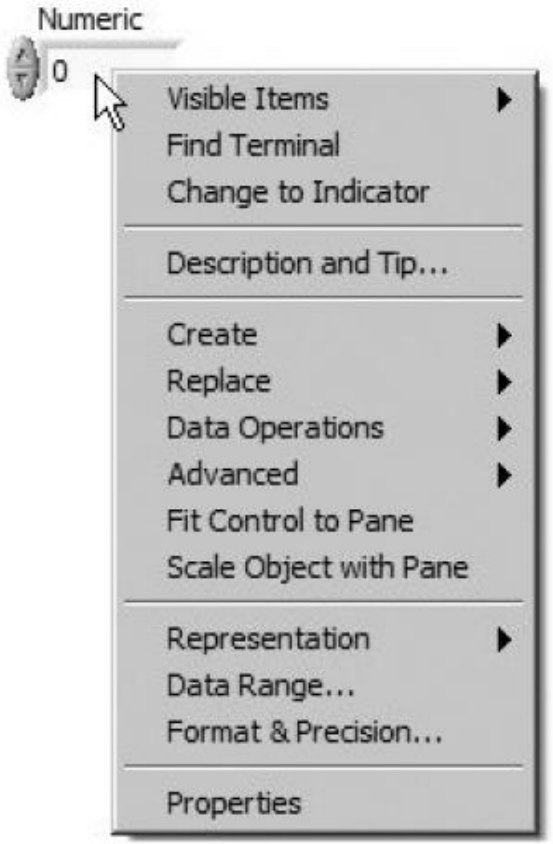
Permet de voir les données qui passent dans le diagramme.

L'environnement LabVIEW

Menus contextuels

Permet d'accéder aux options et commandes disponibles de l'objet en question.

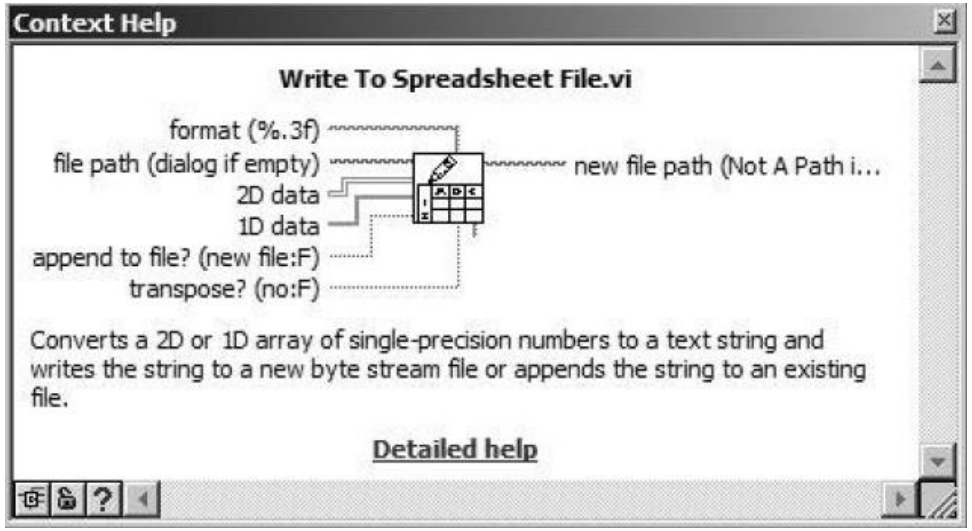
**Explication des différents paramètres p.85*



L'environnement LabVIEW

Aide contextuelle

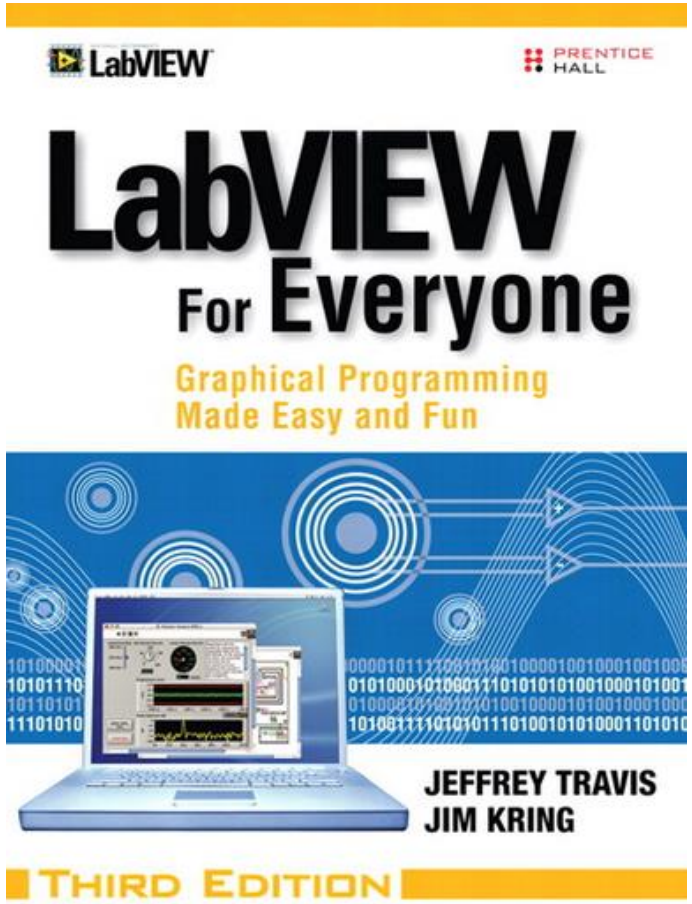
Offre l'information indispensable sur les fonctions, les constantes, les contrôles, les indicateurs, etc… On peut y accéder de différentes façons dont CTRL+H ou encore l'icône jaune avec un point d'interrogation dans la barre d'outil à droite.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

Veuillez vous rendre à la page 94 de votre livre *LabVIEW for Everyone* et compléter l'activité 3-2.



MODULE #1

243-575-RK (3-2-3)

PARTIE 3

LES BASES DE LABVIEW

Enseignant : Sébastien Richard

Les bases de LabVIEW

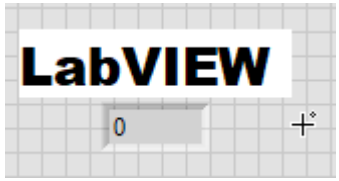
- Découvrir les notions de bases des techniques d'édition LabVIEW
- Découvrir les différents types de contrôles et indicateurs et les options qui s'y rattachent
- Maîtriser les notions nécessaires à la création d'un VI
- Créer et exécuter un VI

Les bases de LabVIEW

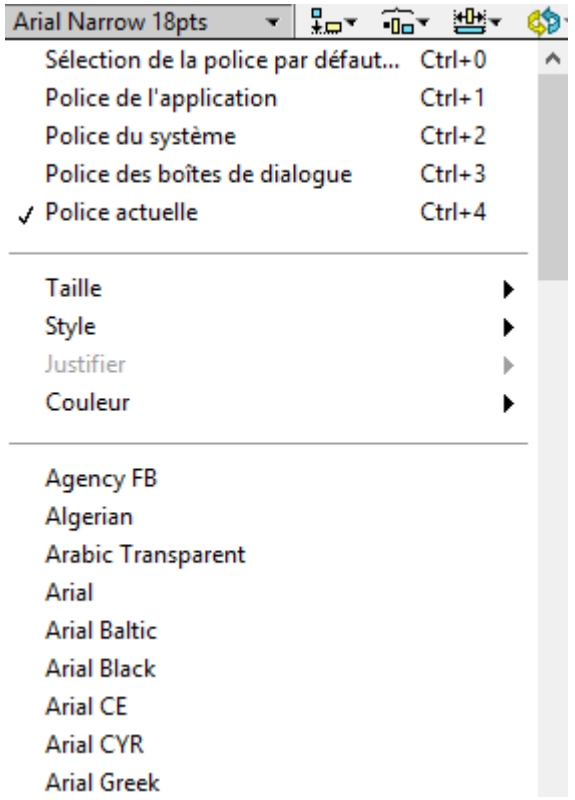
Placer des objets sur le Front Panel

**Lorsque vous placez des objets sur le Front Panel, le terminal correspondant apparaît sur le Block Diagram.*

Étiquetage des objets



Polices, Styles, Taille, Couleur



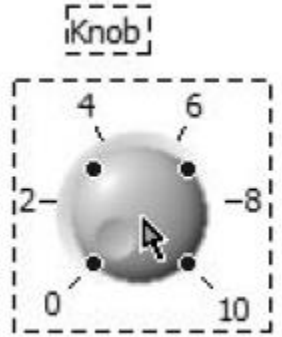
Les bases de LabVIEW

Techniques d'édition

Sélection des objets

Déplacer des objets

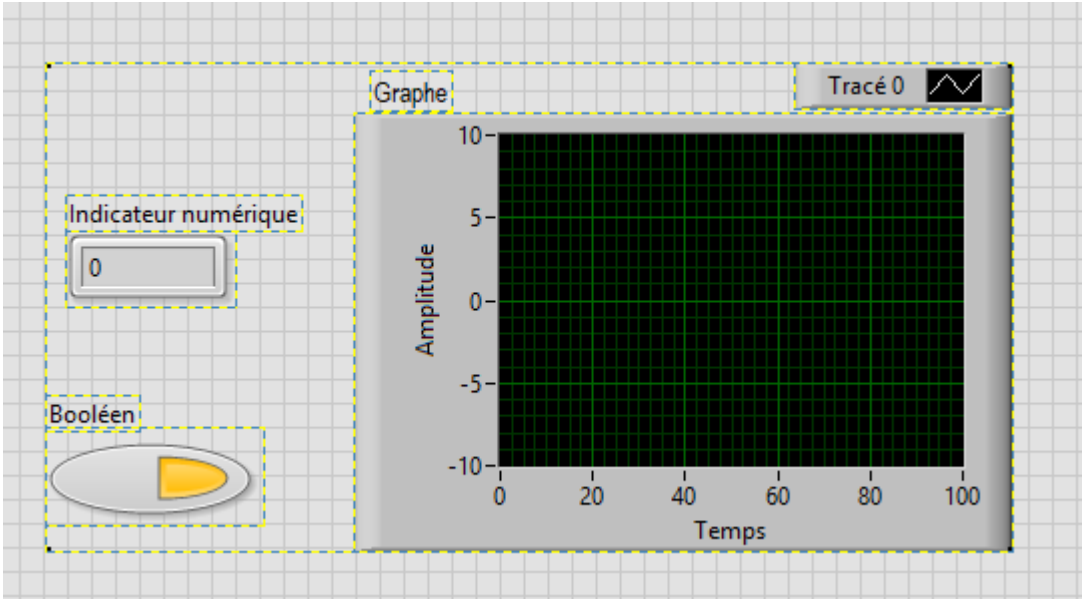
Copier des objets (CTRL… très utile !!)

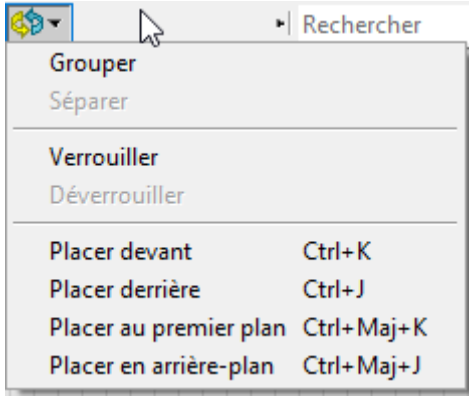


Les bases de LabVIEW

Techniques d'édition

Déplacer, grouper et verrouiller des groupes d'objets.

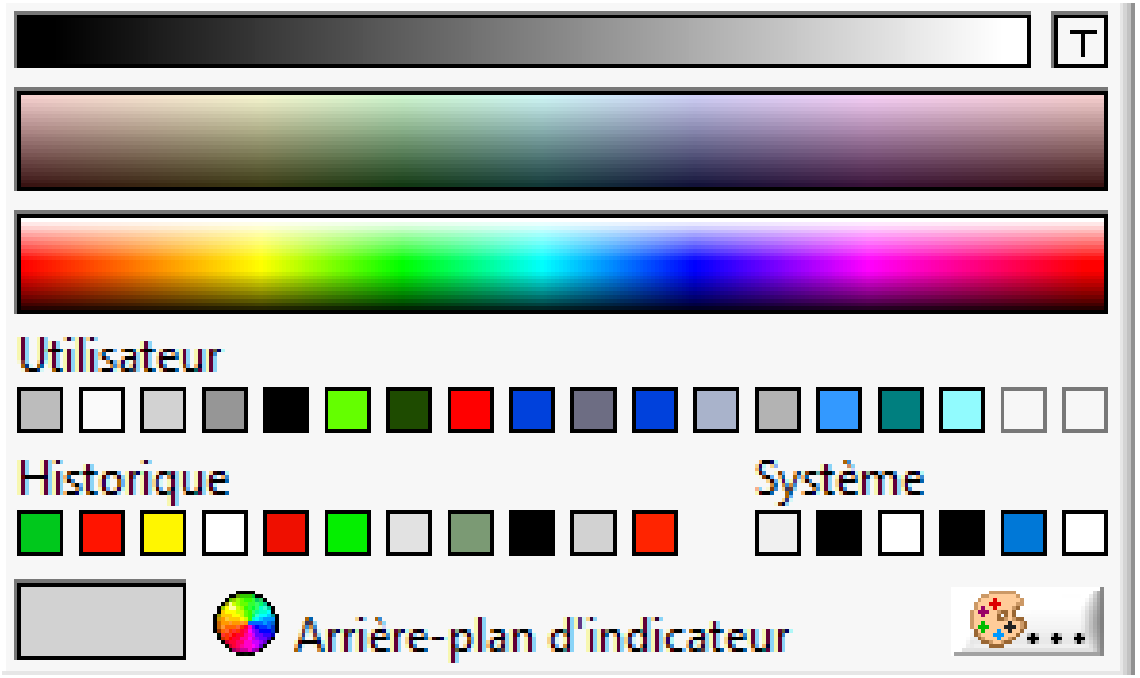




Les bases de LabVIEW

Techniques d'édition

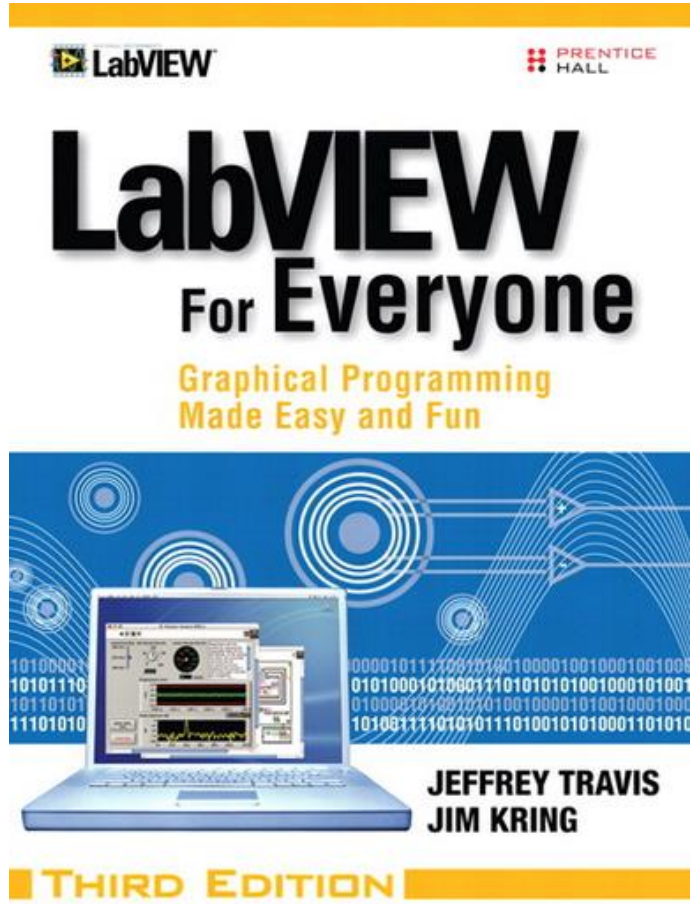
Modification des couleurs



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

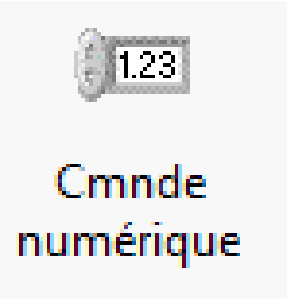
Veuillez vous rendre à la page 113 de votre livre *LabVIEW for Everyone* et compléter l'activité 4-1.



Les bases de LabVIEW – Contrôles et indicateurs

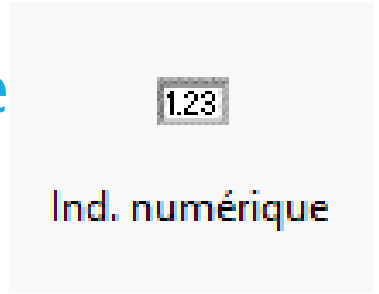
Contrôles numérique:

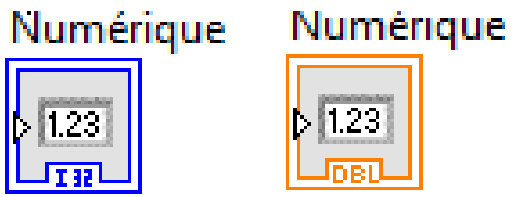
Permet d'entrer des valeurs numériques dans le VI.



Indicateurs numérique:

Permet d'afficher des valeurs numériques que vous voule visualiser.

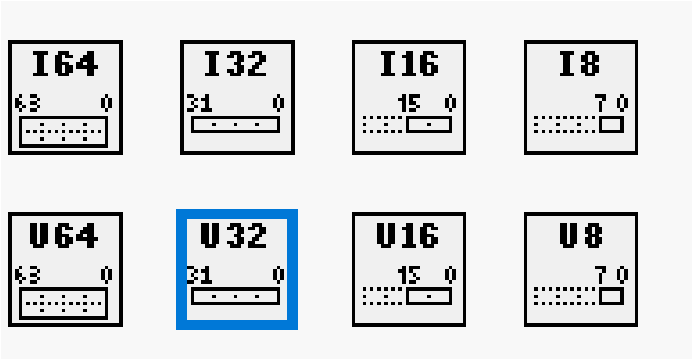




Les bases de LabVIEW - Représentation

L'apparence des terminaux numériques dans le *Block Diagram* dépend de la représentation des données.

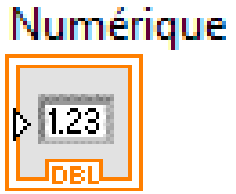
- Nombre de bytes différents pour emmagasiner les données.
- Données peut-être de type « Signed » (Capacité pour les valeurs négatives)
- Données peut-être de type « Unsigned » (Valeur seulement de o à positif)



Les bases de LabVIEW – Représentation

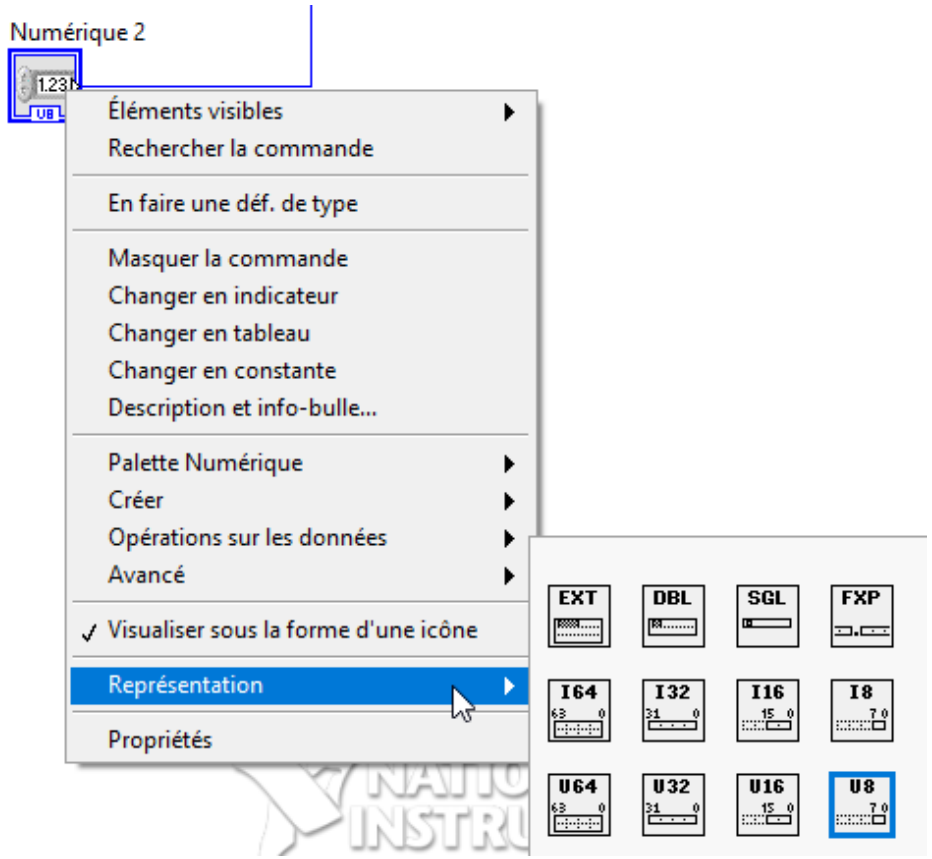
Les terminaux sont de couleur bleue lorsque les données sont de type « Integer ».

Et de couleur orange lorsque les données sont de type « Floating Point ».



Les bases de LabVIEW – Représentation

Vous pouvez changer la représentation des constantes, contrôles ou des indicateurs numériques en affichant le menu de l'objet et en sélectionnant « Représentation »



Les bases de LabVIEW – Représentation

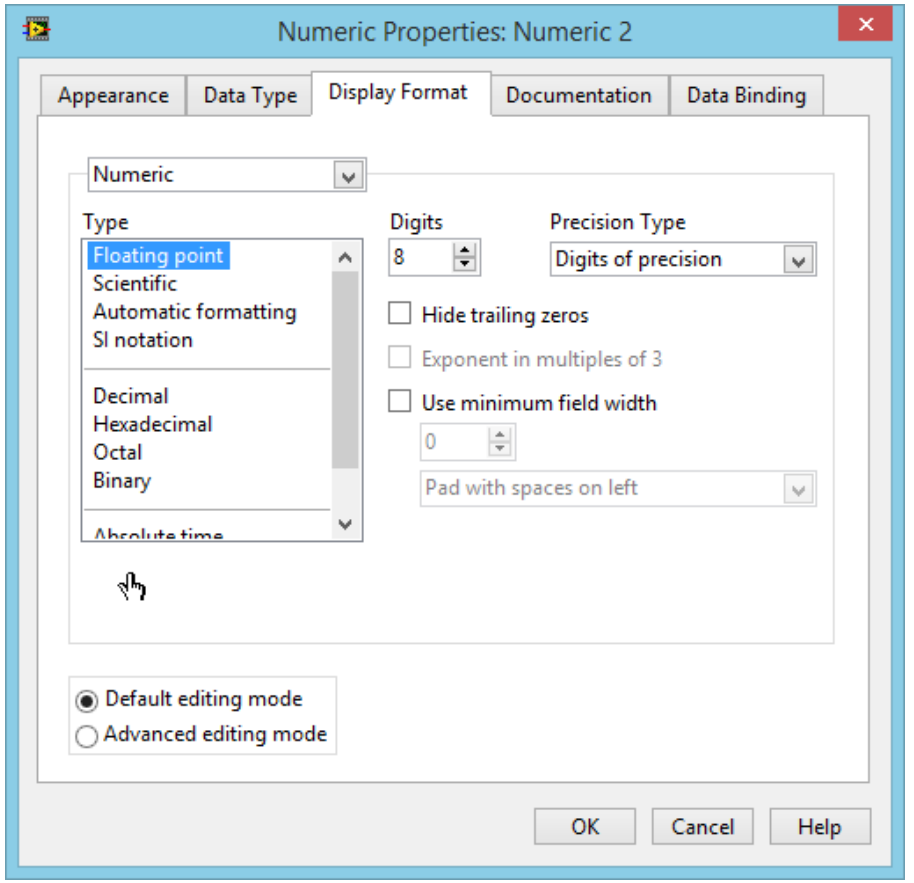
Voici le tableau résumant les différents formats utilissés dans LAbVIEW

Representation	*Abbreviation*	*Terminal (Icon)*	*Terminal*	*Size (bytes)*
byte	I8			1
unsigned byte	U8			1
word	I16			2
unsigned word	U16			2
long	I32			4
unsigned long	U32			4
quad	I64			8
unsigned quad	U64			8
single precision	SGL			4
double precision	DBL			8
extended precision	EXT			10
complex single	CSG			8
complex double	CDB			16

Les bases de LabVIEW – Format et précision

Dans les propriétés de votre contrôle, vous pouvez spécifier le format et la précision que vous voulez donner à l'indicateur.

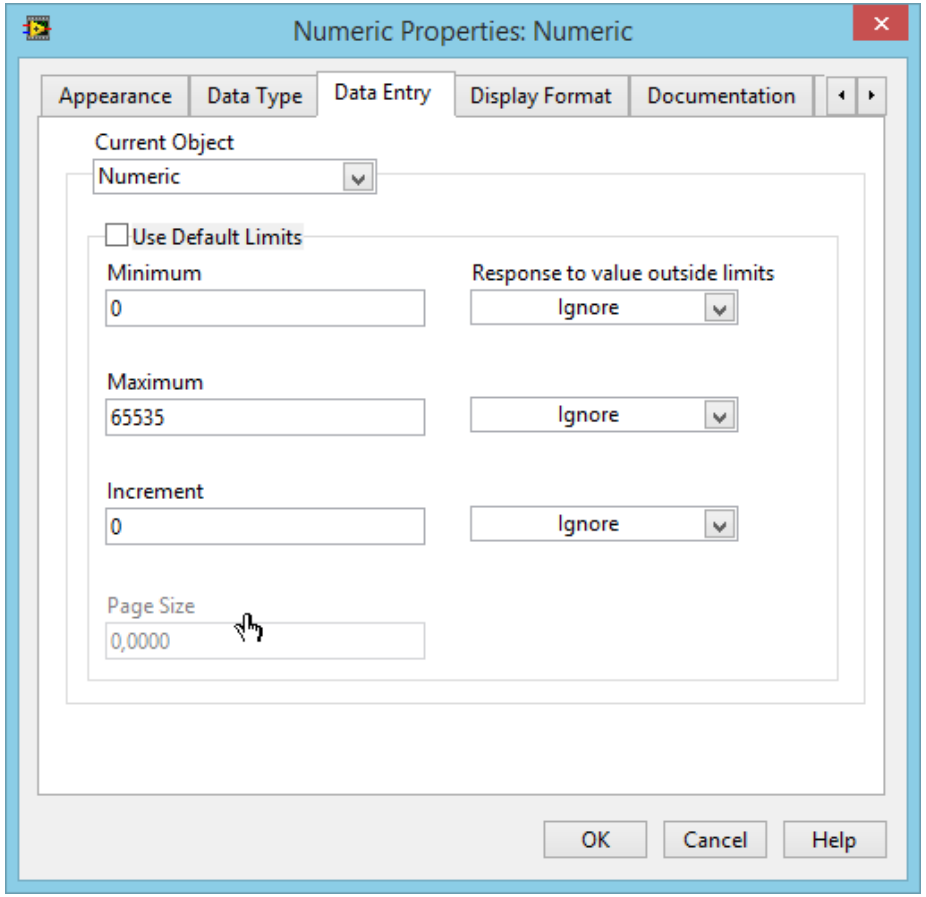
Exemple: Nombre de chiffre après la virgule.



Les bases de LabVIEW – Plage de données

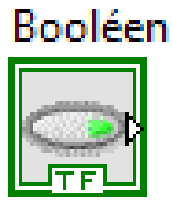
Vous pouvez aussi spécifier la gamme, le « range » valide des données numérique.

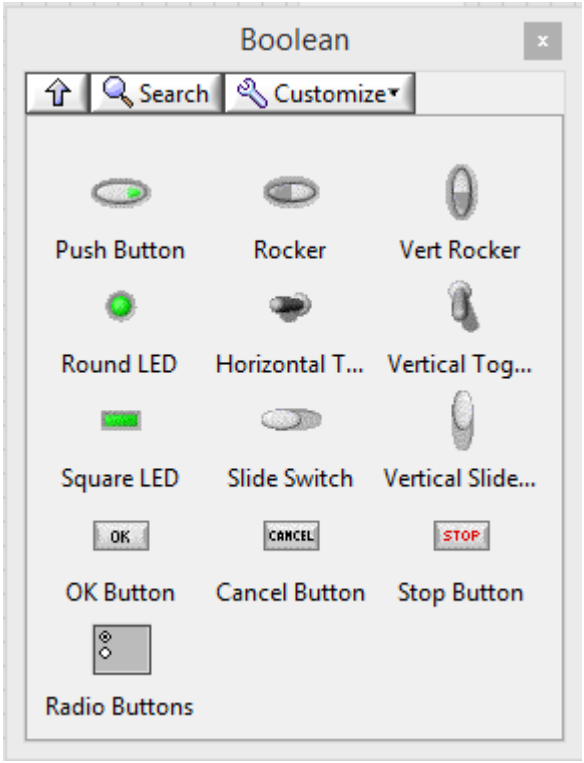
Exemple: Les valeurs minimum et maximum des données qui peuvent être entrées.



Les bases de LabVIEW – Booléen

Le Booléen est tout simplement du « On ou Off ». Les données peuvent avoir un des deux états qui est : **Vrai ou Faux**



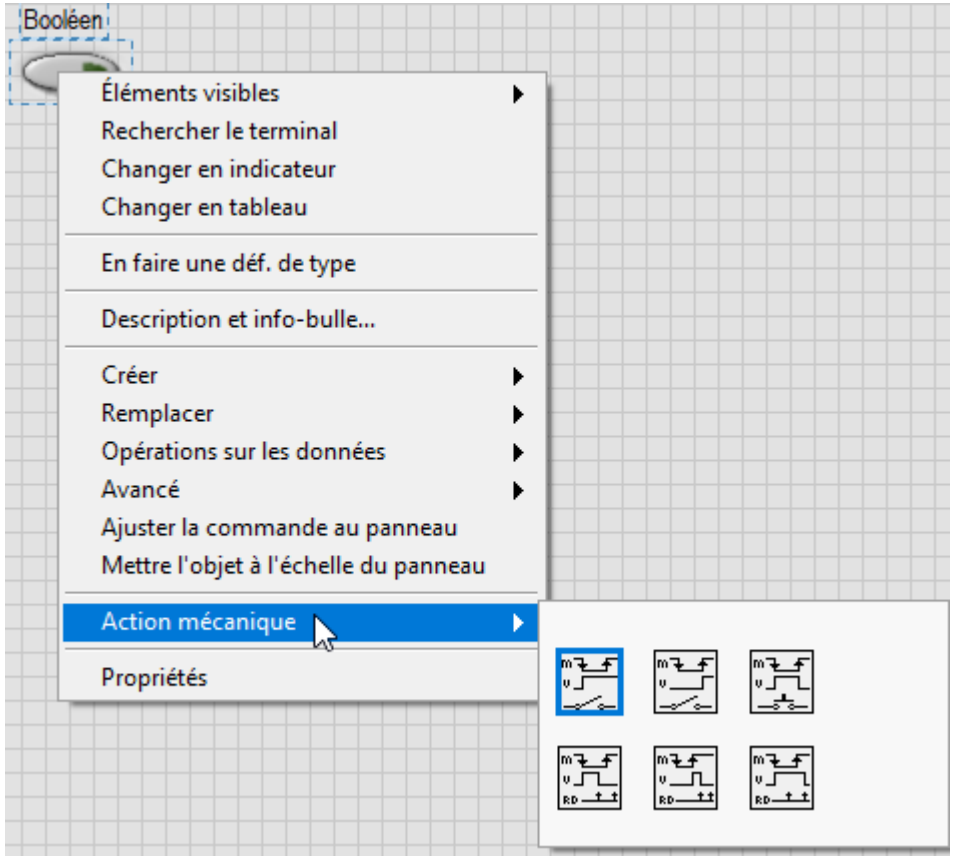


Les bases de LabVIEW – Booléen

Actions mécaniques

(Booléen)

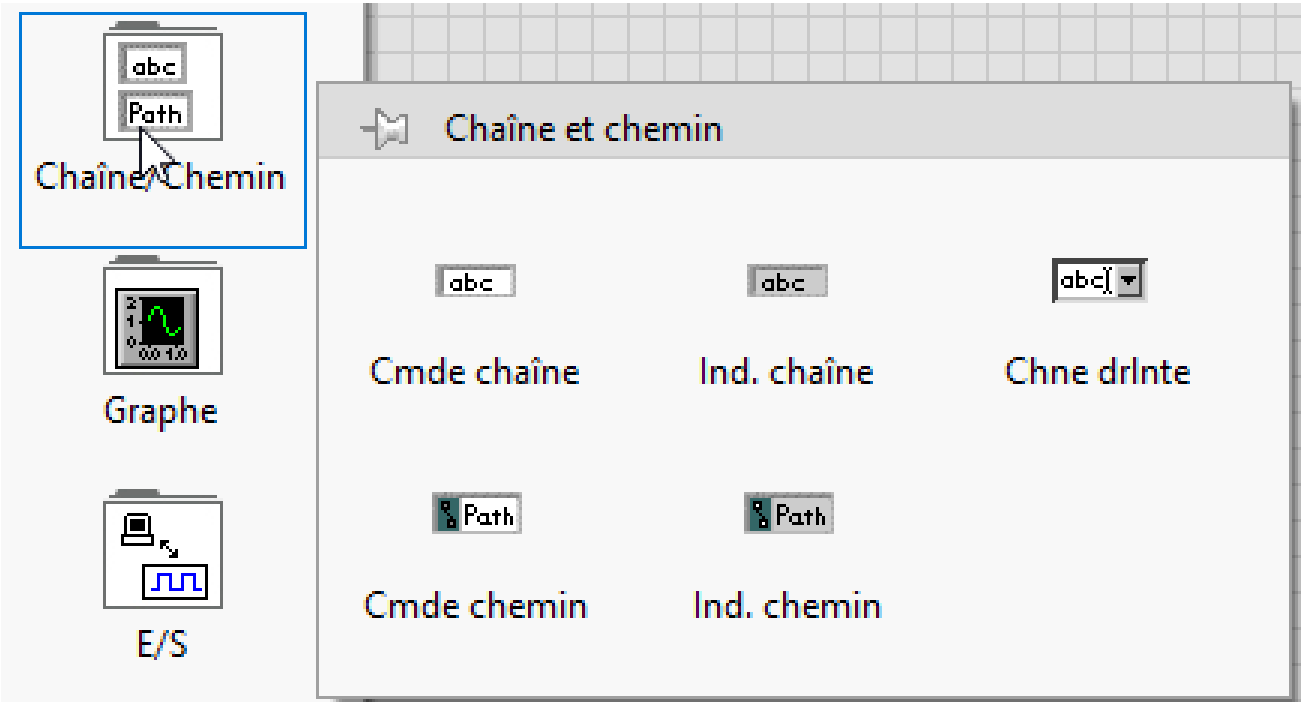
Permet de spécifier comment le contrôle Booléen va réagir quand vous allez appuyer dessus.



Les bases de LabVIEW – Chaîne de caractères

Contrôles et indicateurs (String)

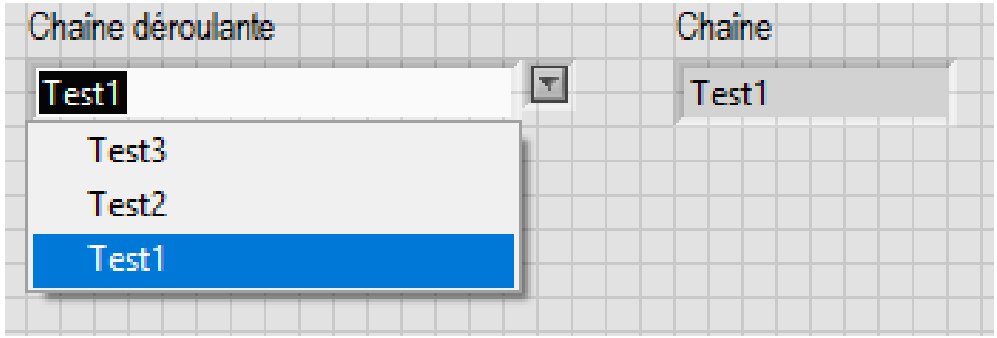
Les contrôles et indicateurs «de type « String » permet d'afficher des données texte.



Les bases de LabVIEW – Chaîne de caractères

Combo box (String)

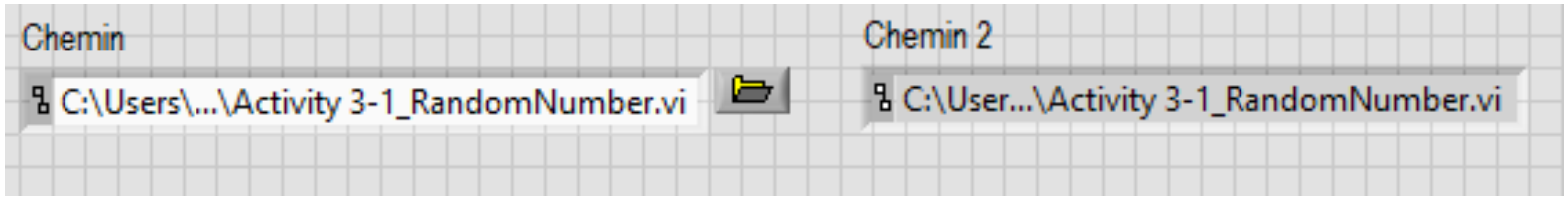
Permet d'afficher des choix que l'on peut éditer dans un menu déroulant.



Les bases de LabVIEW – Chaîne de caractères

Chemin de fichiers/Paths (String)

Permet d'afficher les chemins d'accès aux fichiers ou aux dossiers.



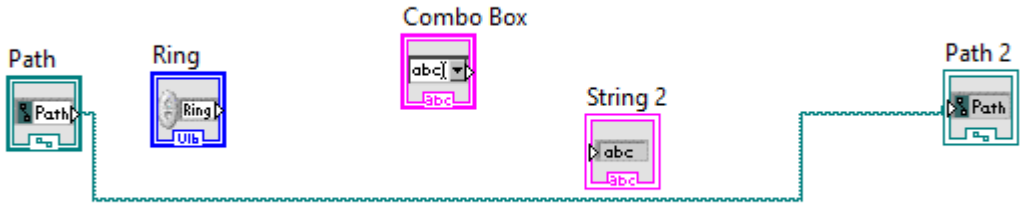
Les bases de LabVIEW – Contrôles et indicateurs

Pour résumé, il y a quatre types de contrôles et indicateurs.

- **Numérique - *numerics*** : Contient les valeurs numériques standards
- **Booléen - *boolean*** : Ne peux avoir que deux états possibles, vrai ou faux.
- **Chaîne de caractères - *string*** : Contient les données de texte. Même si elle peut contenir des caractères numériques (de 0 à 9), vous devez convertir cette donnée *string*, en donnée *numérique* si vous désirez faire des opérations arithmétiques avec cette donnée.
- **Chemins d'accès – *paths*** : Type de donnée utile pour tout ce qui est de la gestions et l'affichage des chemins d'accès.

Les bases de LabVIEW – Routage

Le routage auto permets de compléter les liens en évitant les obstacles dans votre *Block Diagram* dans :



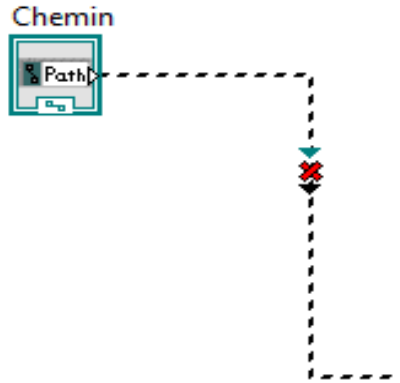
Malgré que cette function semble utile, certaines personnes ne l'aime pas. Si vous voulez active ou désctiver cette function,

Tools>Options>>Block Diagram>>"Enable automatic wire routing").

Les bases de LabVIEW – Routage

Mauvais câblage

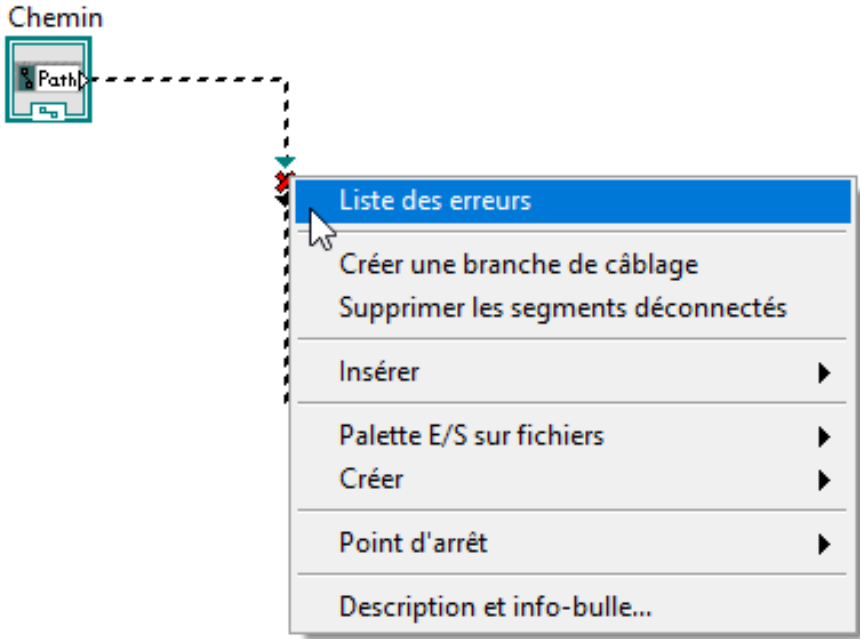
Lorsque le câble est brisé avec un X, vous pouvez toujours le supprimer avec le raccourci Ctrl-B.

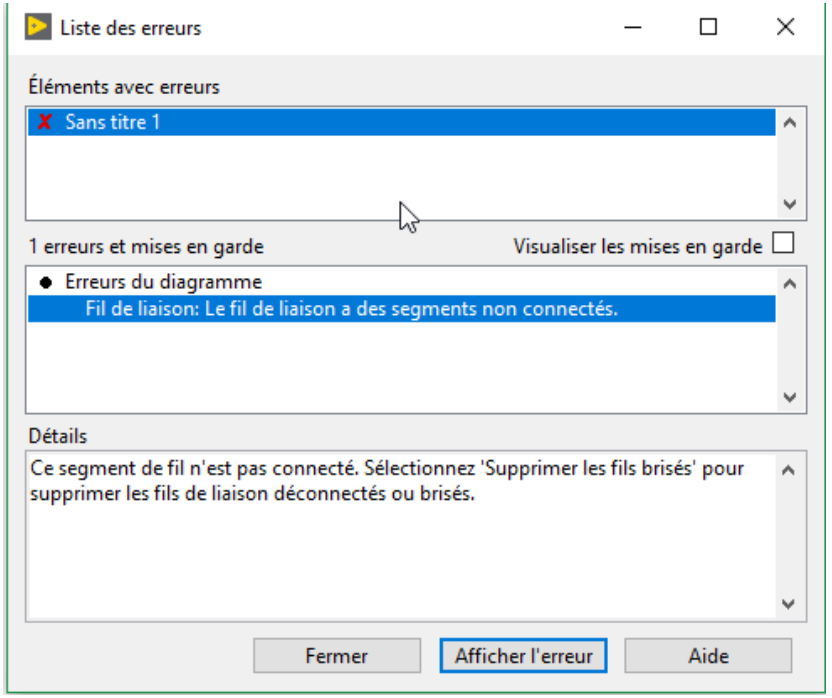


Les bases de LabVIEW – Routage

Mauvais câblage

Vous pouvez aussi vérifier le pourquoi, la raison de cette erreur en sélectionnant la Liste des erreurs.

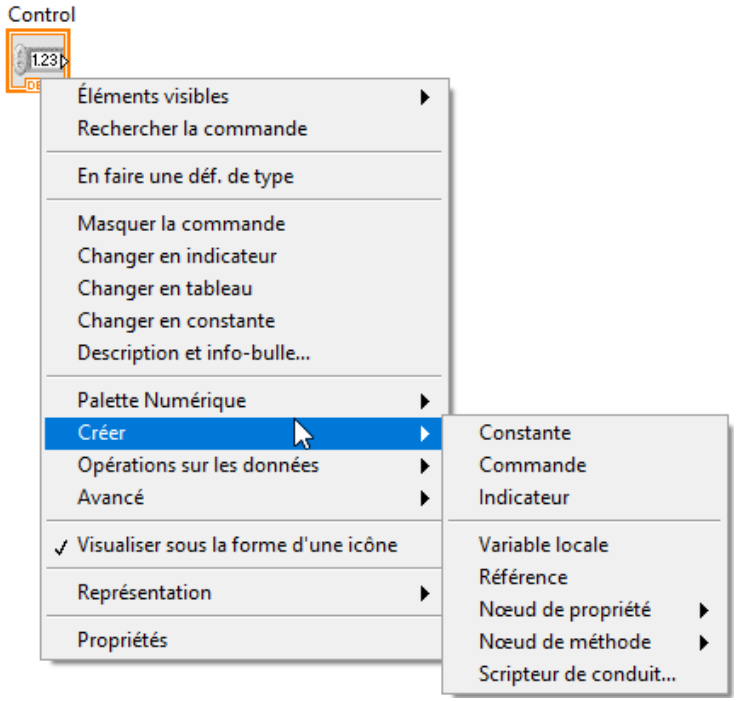




Les bases de LabVIEW – Ajout

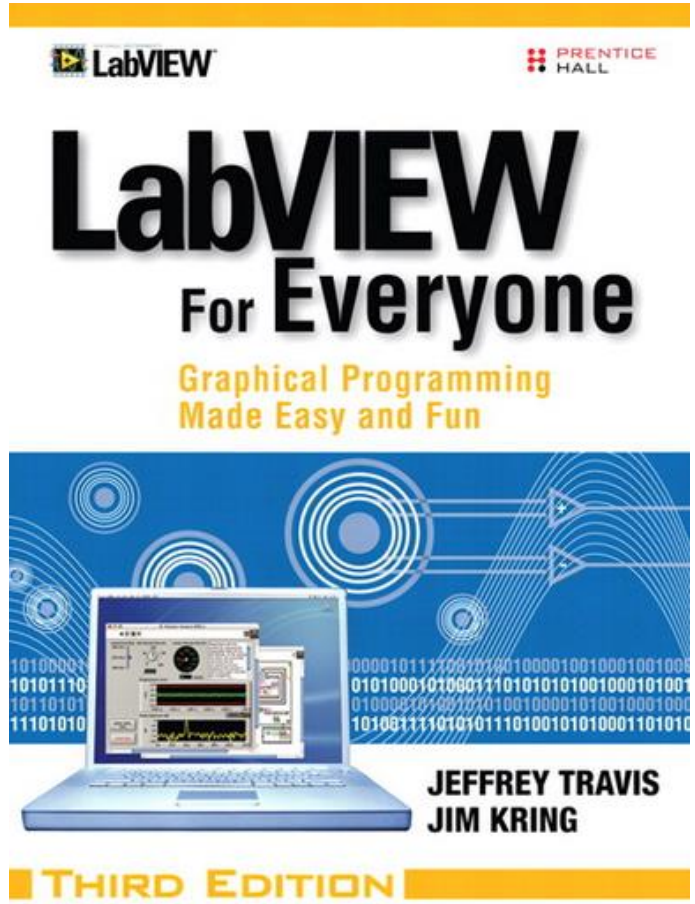
Constantes, Contrôles et indicateurs

Il est possible d'ajouter un terminal automatiquement selon l'objet sur lequel vous le demandez sans passer la palette de contrôles.



L'environnement LabVIEW – Trucs pratiques

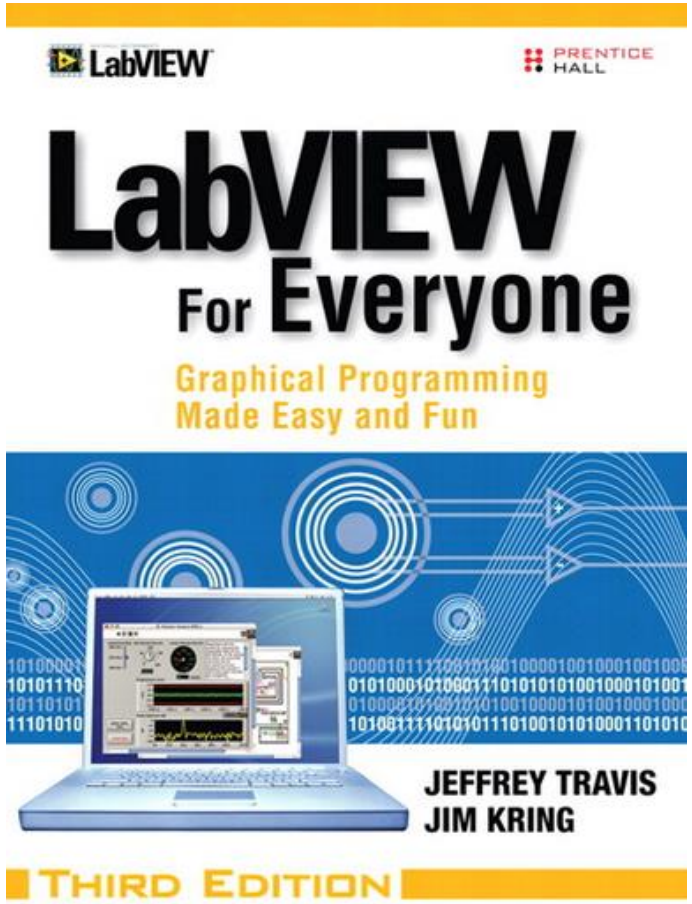
Lisez et pratiquez les différents petits trucs donnés à la page 140 à 143 de votre livre LabVIEW for Everyone, dans la section *Useful Tips*.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

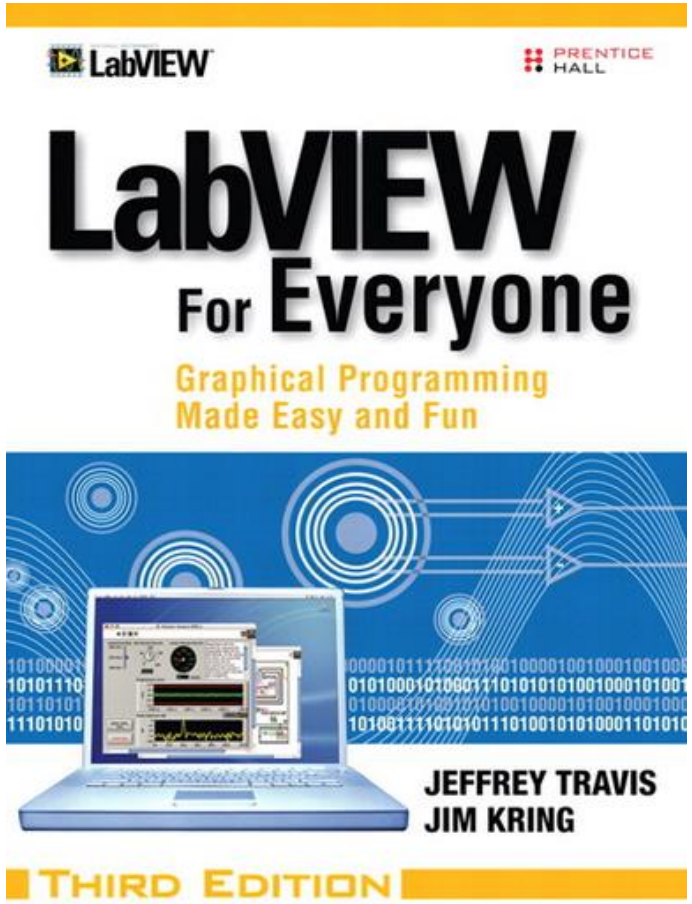
Veuillez vous rendre à la page 137 de votre livre *LabVIEW for Everyone* et compléter l'activité 4-2.



L'environnement LabVIEW – Pratique

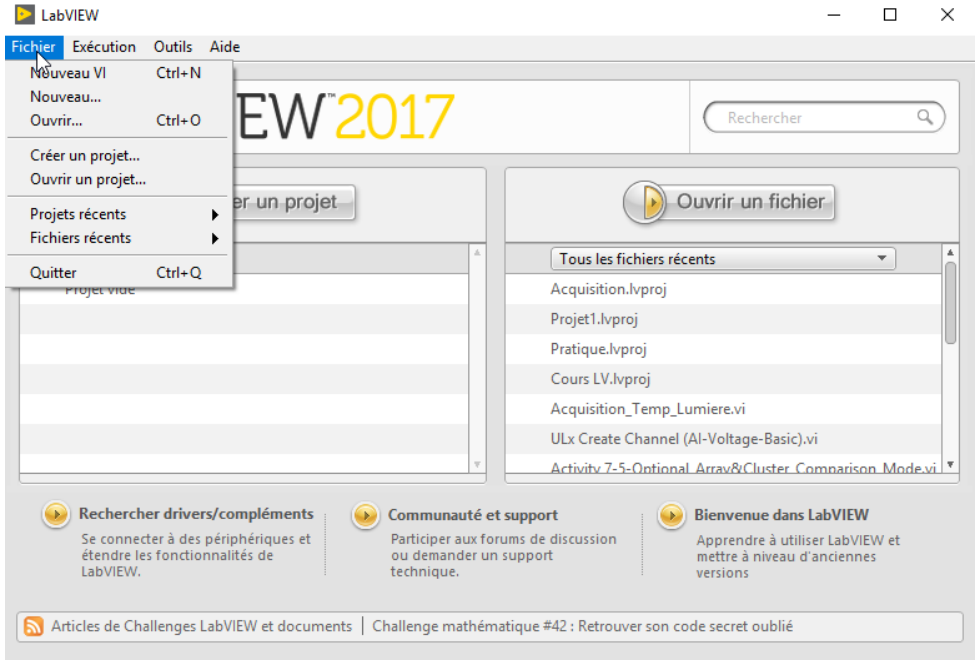
C'est le temps de passer à la pratique! **Survol des fonctions les plus importantes**

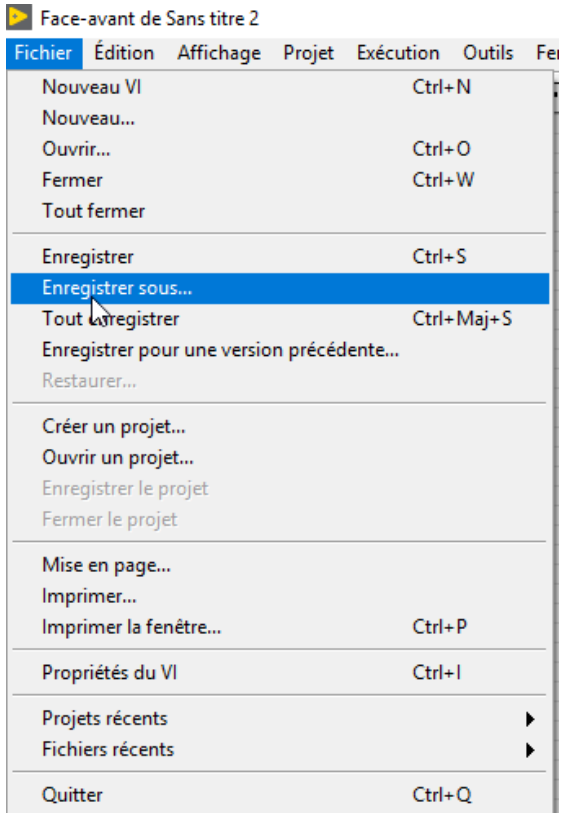
Veuillez vous rendre à la page 145 de votre livre *LabVIEW for Everyone* et compléter l'activité 4-3 et 4-4.



Les bases de LabVIEW – Sauvegarde

Charger et sauvegarder vos VIs

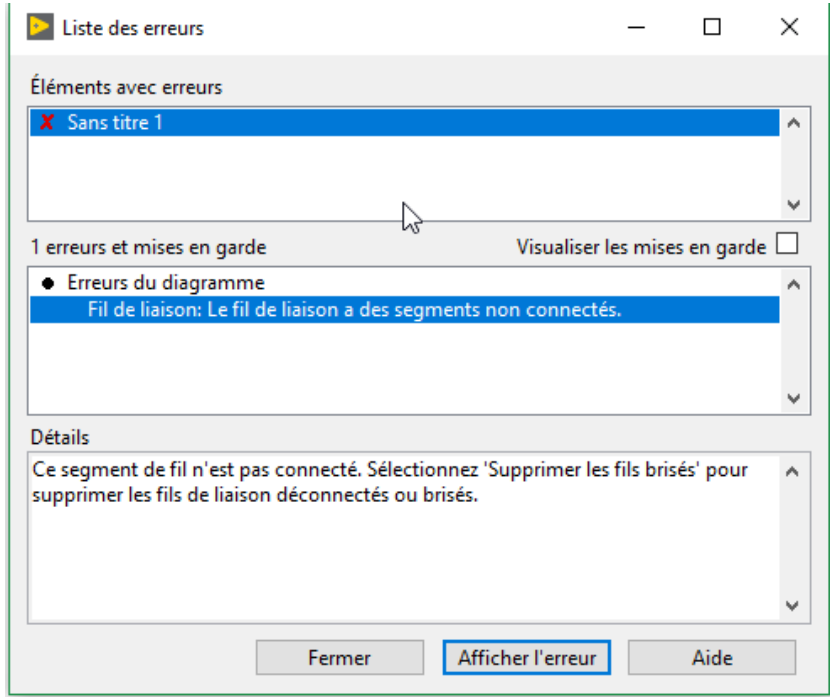




Les bases de LabVIEW – Débogage

Utiliser les outils de débogage de LabVIEW

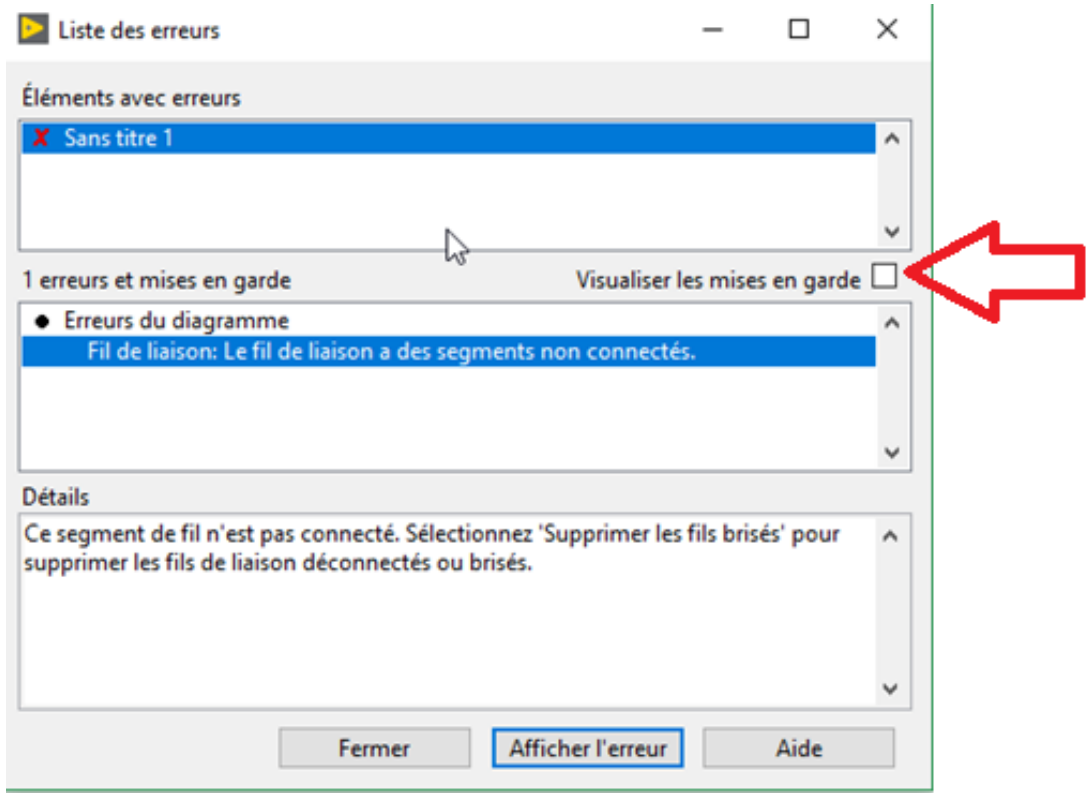
Vous pouvez consulter la liste des erreurs afin de comprendre pour quelle raison votre VI ne compile pas, ne démarre pas. Vous pouvez aussi localiser l'erreur dans votre programme G à l'aide du bouton Afficher l'erreur.



Les bases de LabVIEW – Débogage

Utiliser les outils de débogage de LabVIEW

Il y a aussi les Avertissements. Vous pouvez visualiser les avertissements en cochant la case « Visualiser les mises en garde »,

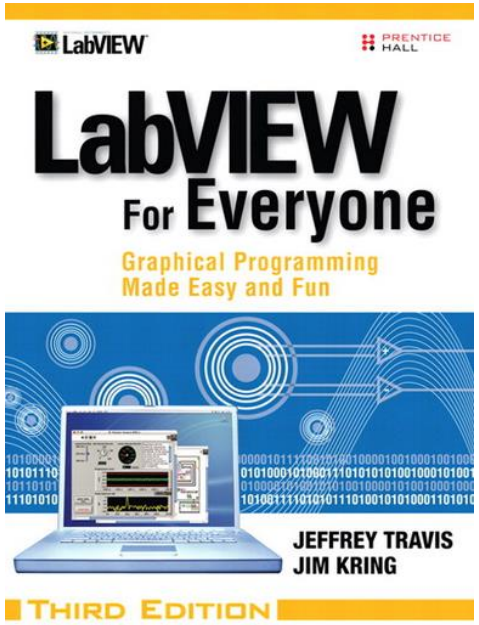


Les bases de LabVIEW – Débogage

Erreurs les plus fréquentes

Une liste des erreurs les plus fréquentes est donnée à la page 155.

**Ceci vous donnera un bon coup de main afin de mieux comprendre les erreurs de compilation.*

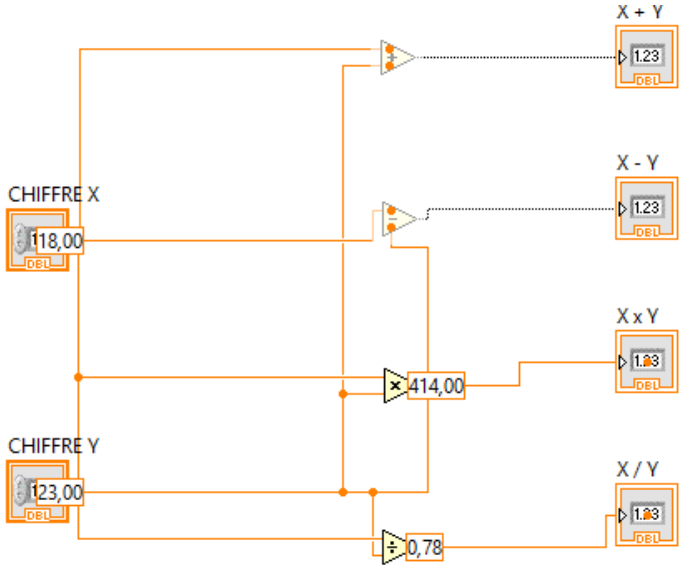


Les bases de LabVIEW – Débogage

Execution Highlighting

Il est souvent utile de voir exactement ce qui arrive avec nos données dans le *Block Diagram*. Le mode d'exécution pas à pas est un outil indispensable pour votre débogage et voire même votre meilleur allié afin de réussir à comprendre le trajet de vos données et le fonctionnement de vos programmes.

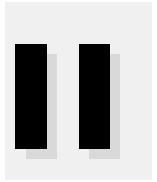
Pour utiliser ce mode, appuyer sur 



Les bases de LabVIEW – Débogage

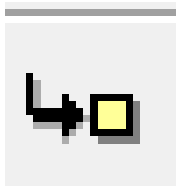
Bouton Pause

Ajouter la fonction Pause à la fonction Execution Highlight et vous serez en mesure de bien visualiser, mettre en pause votre programme à une section particulière, afin de voir le fonctionnement de votre VI et le trajet de vos données.



Bouton Étape par étape

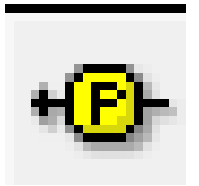
Ajouter la fonction Single-Step à la fonction Execution Highlight et vous serez en mesure de bien visualiser, étape par étape le fonctionnement de votre VI et le trajet de vos données.



Les bases de LabVIEW – Débogage

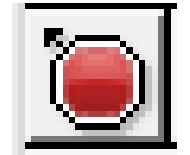
Sonde

L'utilisation de la sonde permet de vérifier les valeurs des données qui passent dans les fils.



Break Point

Cet outil permet de suspendre l'exécution à un point précis de votre programme.



Les bases de LabVIEW – Débogage

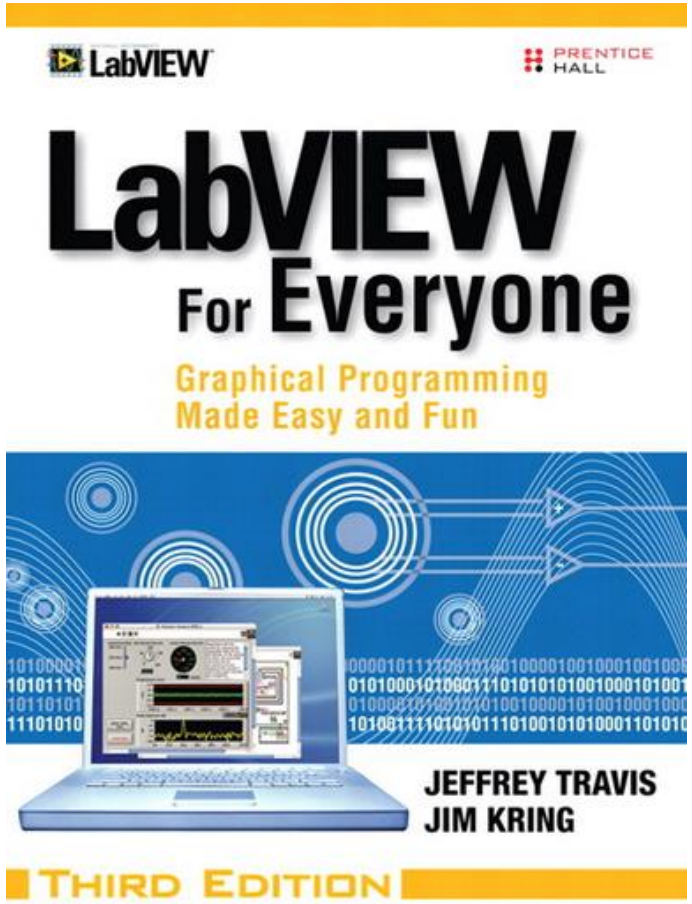
En résumé, pour un bon diagnostique, vous devez être en mesure d'utiliser tous les outils de débogage présentés.

Ces outils, sont encore plus efficaces lorsqu'ils sont utilisés en combinaison afin de bien localiser le problème.

L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

Veuillez vous rendre à la page 160 de votre livre *LabVIEW for Everyone* et compléter l'activité 5-1.



L'environnement LabVIEW – Le SubVI

Un Sous-Vi est tout simplement un Vi utilisé dans un autre VI ou encore, demandé par un autre Vi.

Vous pouvez construire votre programme un module à la fois en créant des Sous-Vi(s).

Cela permet ainsi de simplifier la compréhension de votre programme principal en créant des icones comportant des entrées et sorties fonctionnant selon le programme inséré.

L'environnement LabVIEW – Le SubVI

Création d'un Sous-Vi

Avant d'utiliser un Vi comme étant un Sous-Vi, il faut le préparer.

Deux éléments doivent être configurés :

- La création d'un icône
- L'assignation des connecteurs.

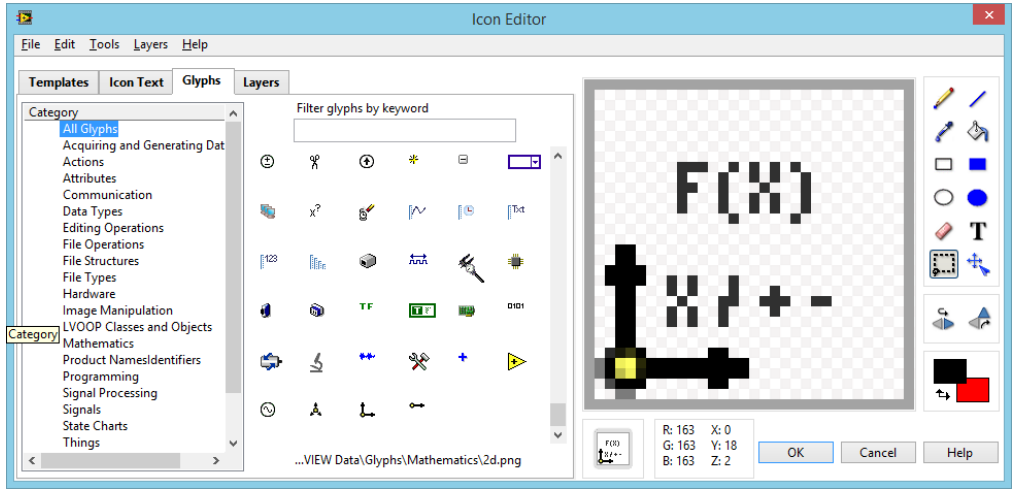
Il faudra donc lui assigner un icône qui sera représentatif de sa fonction et de bien configurer les connecteurs afin de pouvoir l'interconnecter efficacement dans un autre programme.

L'environnement LabVIEW – Le SubVI

1. Création de l'icône

Après avoir double-cliqué sur l'icône dans le coin supérieur droit de votre block diagram, l'éditeur d'icône vous apparaîtra.

Modifier l'icône avec le outils de dessin afin que celui-ci donne une bonne indication de la fonction de votre SubVI.



L'environnement LabVIEW – Le SubVI

2. Assignation des connecteurs

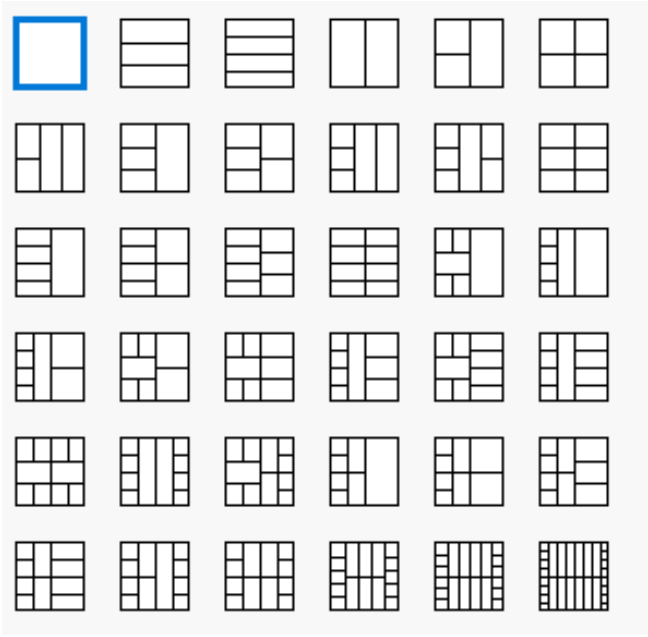
Pourquoi assigner des connecteurs ? Afin de pouvoir faire passer des données à l'entrée et à la sortie (IN et OUT) du Sous-Vi.

** Une bonne pratique est de sélectionner des connecteurs d'entrées à la gauche et des connecteurs de sorties à la droite.*

L'environnement LabVIEW – Le SubVI

Assignation des connecteurs

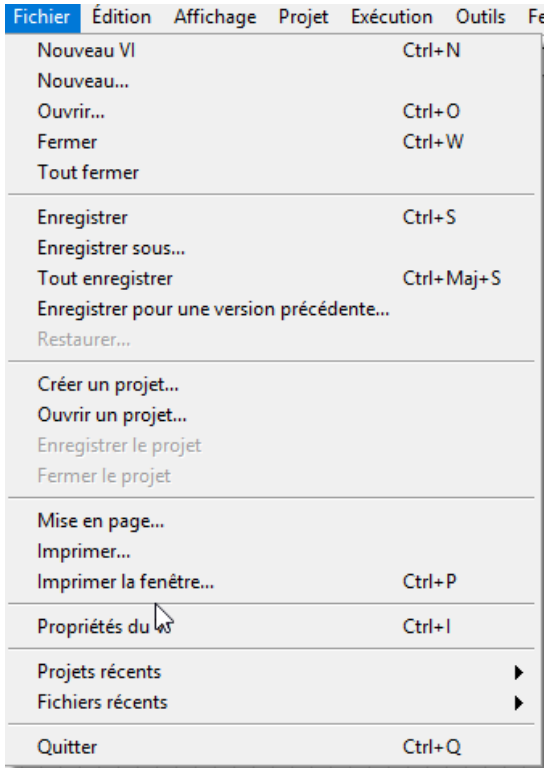
Vous devez sélectionner le bon bloc, c'est-à-dire celui avec le bon nombre d'entrées et de sorties.

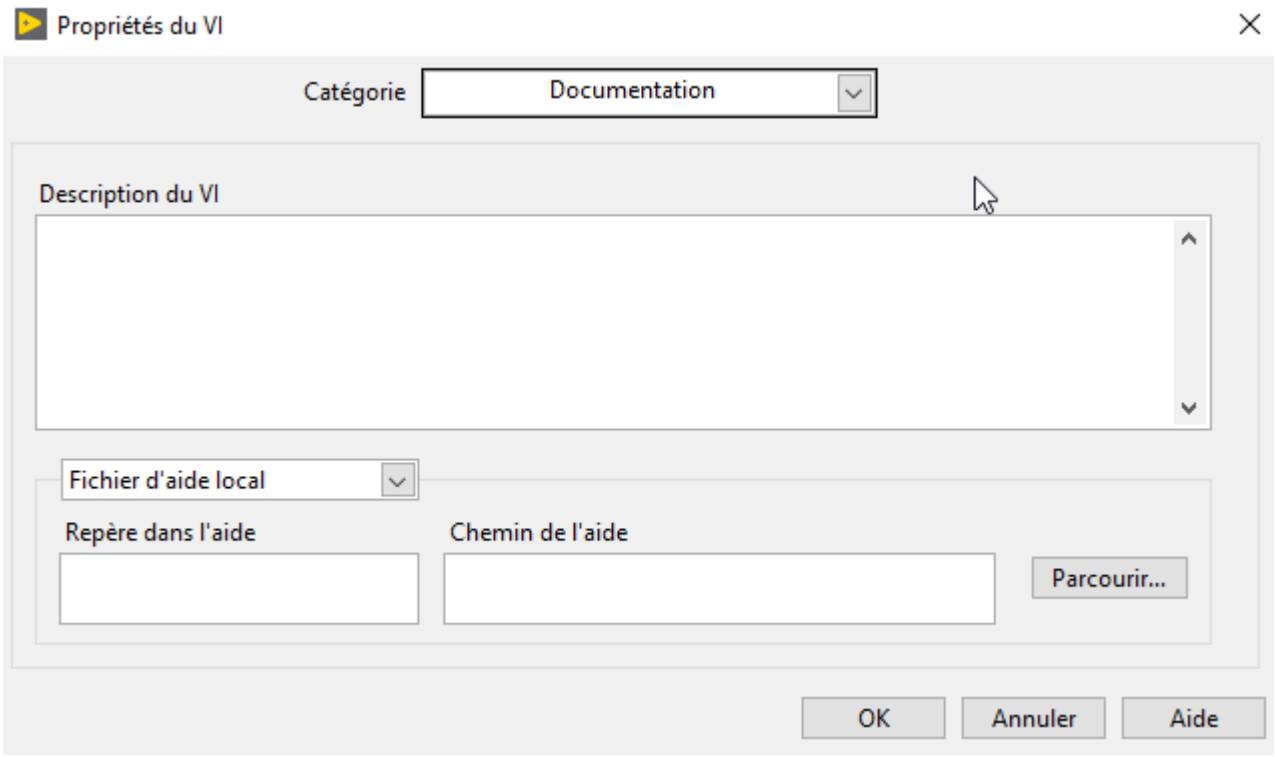


L'environnement LabVIEW – Le SubVI

Documenter un VI

Vous pouvez aussi documenter vos VI.

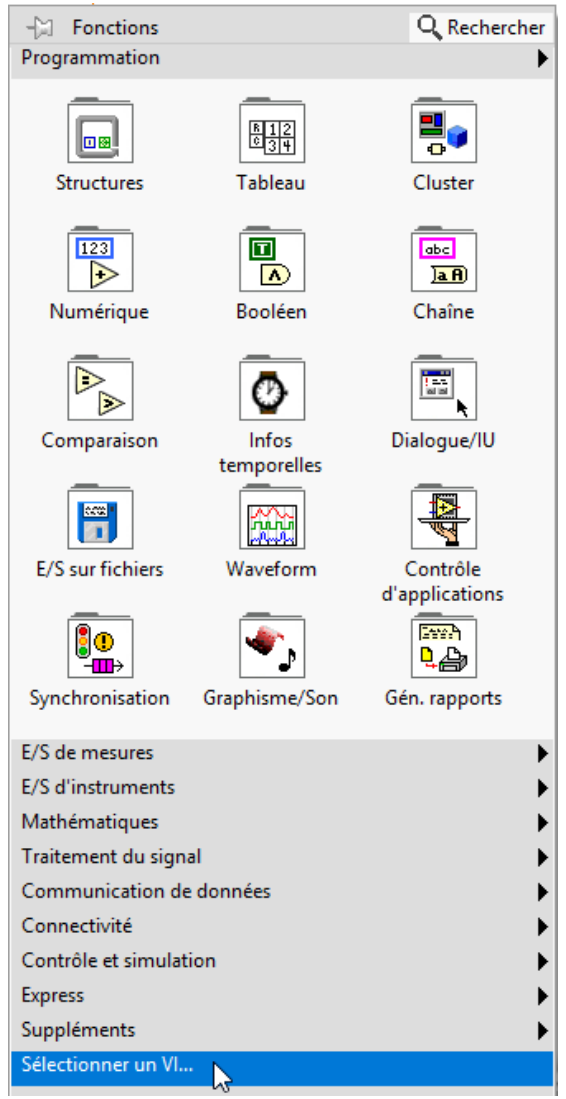




L'environnement LabVIEW – Le SubVI

Sélection de Sous-Vi

Pour utiliser un sous-Vi dans un Vi, il suffit de choisir l'option « Sélectionner un Vi » dans le Block Diagram.



L'environnement LabVIEW – Le SubVI

Exercice Création d'un Sous-Vi.

À faire:

- Ajouter 4 contrôles numériques (IN)
- Concevoir une fonction qui indique le plus grand nombre des 4 et qui le retourne.
- Ajouter un indicateur numérique (OUT) qui affiche le résultat de la fonction.

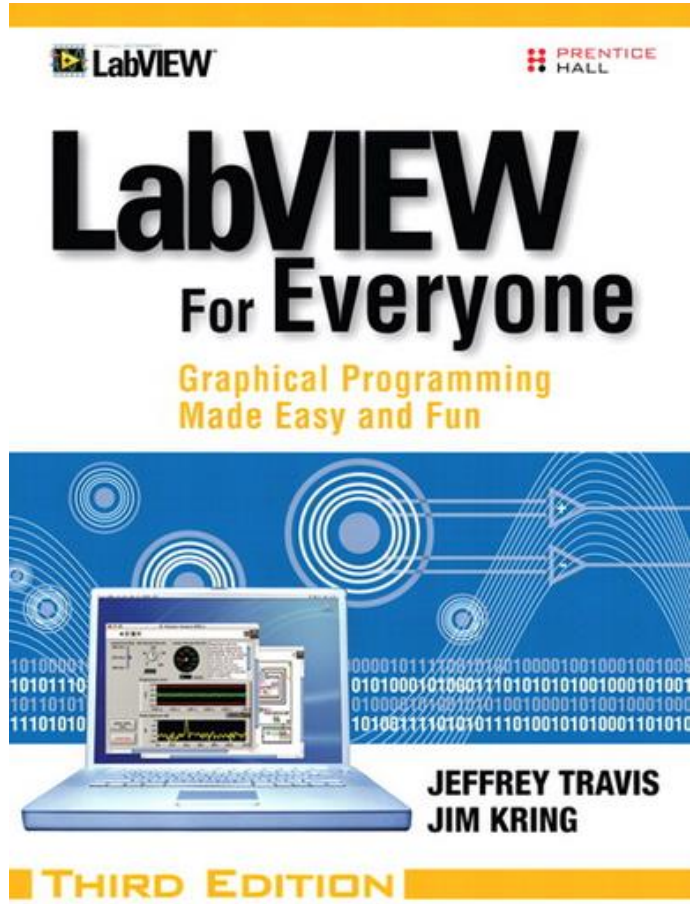
**Faire rouler votre Vi afin de valider le bon fonctionnement.*

Lorsque le tout fonctionne correctement, veuillez vous créer un Sous-Vi avec ce VI en suivant les étapes de la p.164 à 175.

L'environnement LabVIEW – Pratique

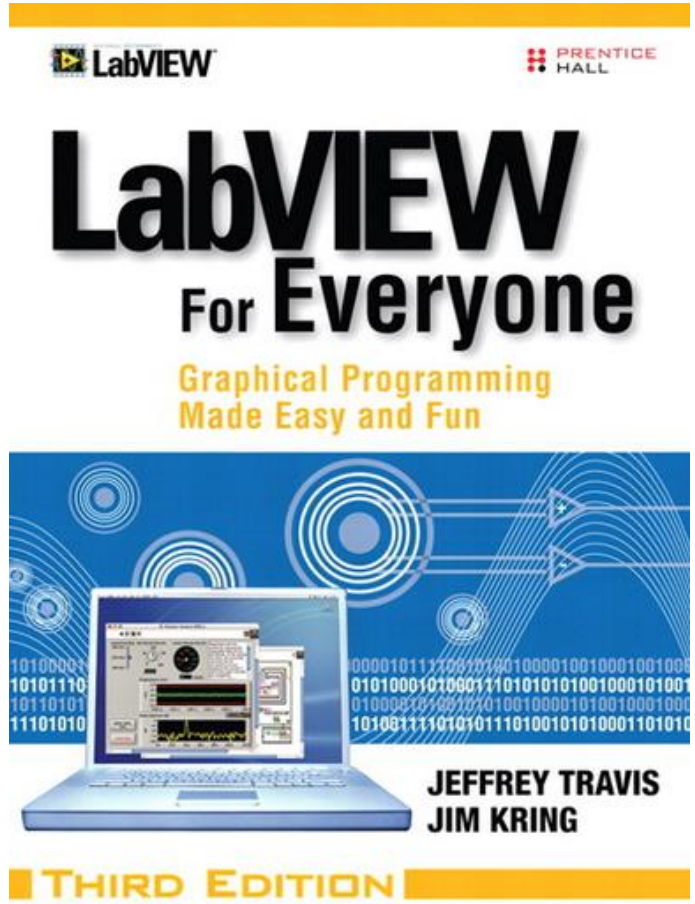
C'est le temps de passer à la pratique!

Veuillez vous rendre à la page 178 de votre livre *LabVIEW for Everyone* et compléter les activités 5-2, 5-3 et 5-4.



L'environnement LabVIEW – Pratique

L'exercice suivant ne se trouve pas dans le livre. Il suffit de faire un programme qui fait la conversion d'un nombre décimal en hexadécimal et aussi en binaire. Vous devez vous assurez de limiter la plage d'entrée de votre contrôle afin de ne pas dépasser la capacité de votre convertisseur.



Contrôle de l'exécution des programmes

- Connaître et différencier les boucles *For* et *While*
- Connaître les différents types de *Case*
- Utiliser les *Formula Nodes* pour les calculs complexes
- Utiliser les fenêtres de dialogue
- Comprendre l'utilisation des outils de timing
- Apprendre à combiner les boucles *While* et les structures *Case* comme base d'architecture.
- Connaître les registres à décalage (Shift Registers)

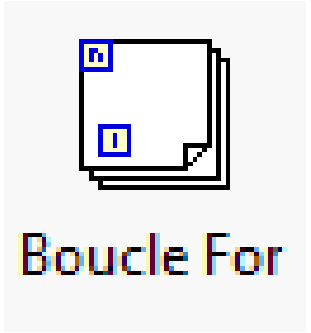
Boucle *FOR* et *WHILE*

Deux types de boucles sont utilisés dans LabVIEW afin de répéter certaines opérations dans votre programme.

La boucle FOR répètera cette portion de programme pendant un nombre spécifique de fois.

La boucle WHILE, la répètera tant que la condition sera VRAI ou FAUX selon notre

Contrôle de l'exécution des programmes – Boucle For



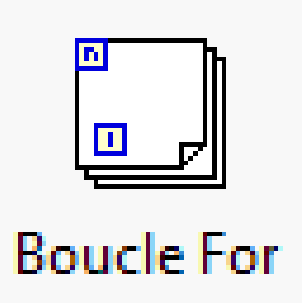
La boucle *For*

Une boucle *For* exécute le code à l'intérieur de sa boucle pour un total de X fois qui est la valeur contenue dans le « Count Terminal », le N.

Contrôle de l'exécution des programmes – Boucle For

La boucle *For*

Le terminal d'itération (I) contient le nombre à jour d'itération, de boucle complété.



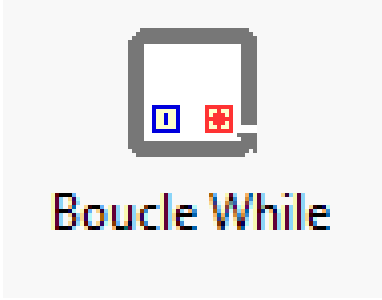
Bien important : le nombre d'itération débute à 0. Donc pendant la première itérations le I = 0, pendant l'exécution de la deuxième le I est = 1

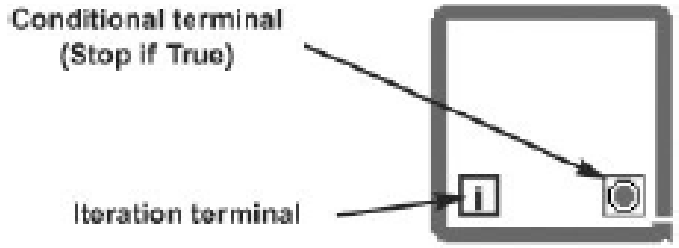
Donc la boucle se termine à N-1.

Contrôle de l'exécution des programmes – Boucle While

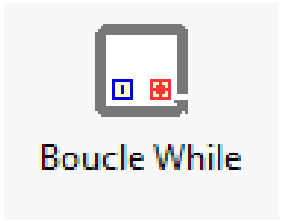
La boucle *While*

Une boucle While exécute le sous-diagramme à l'intérieur de sa boucle « Tant que » la valeur Booléen câblé à son terminal de condition soit VRAI ou FAUX dépendamment comment l'on configure le terminal.





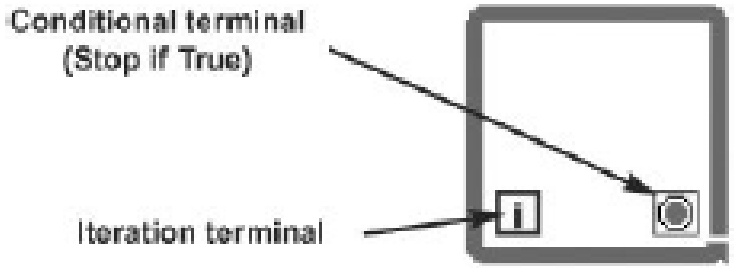
Contrôle de l'exécution des programmes – Boucle While

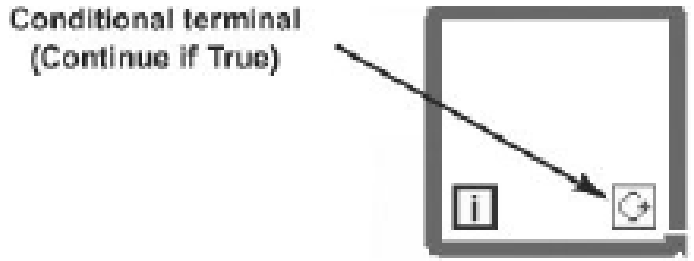


La boucle *While*

LabVIEW vérifie la valeur du terminal de condition à la fin de chaque itération et n'arrête pas tant que la condition n'est pas respectée. La condition est soit:

Arrêter si VRAI ou **Continuer si VRAI**

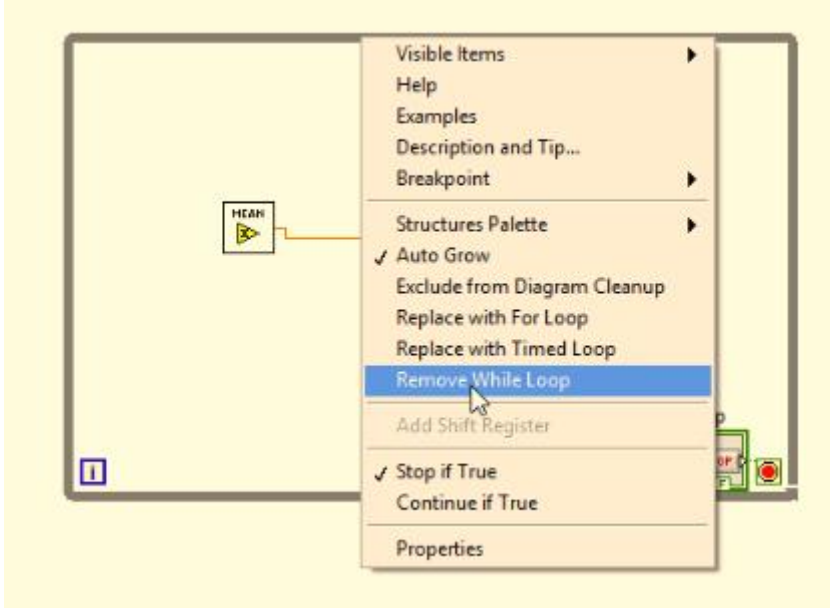




Contrôle de l'exécution des programmes – Boucle While

La boucle While

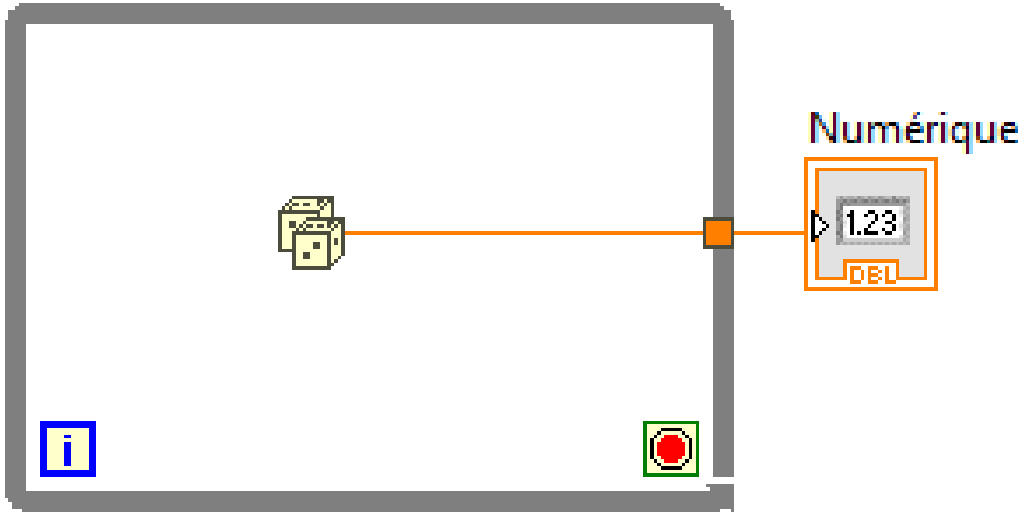
Lorsque vous voulez effacer une boucle sans perdre son contenu vous devez sélectionner « Remove While Loop » à partir du menu contextuel de la boucle elle-même.



Contrôle de l'exécution des programmes – Tunnel

Tunnel

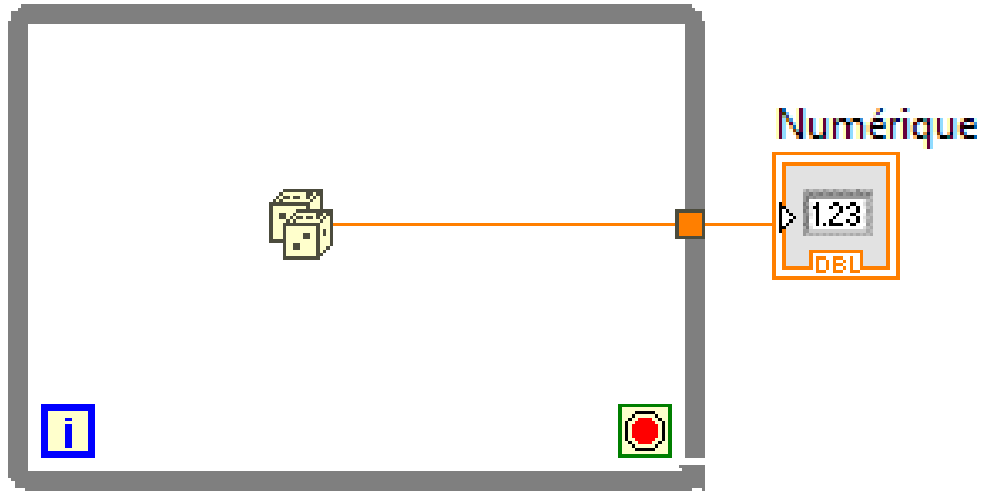
Les données passent à l'intérieur et à l'extérieur de la boucle par une petite boîte aux limites de la boucle que l'on nomme : Tunnel.



Contrôle de l'exécution des programmes – Tunnel

Dataflow / Tunnel

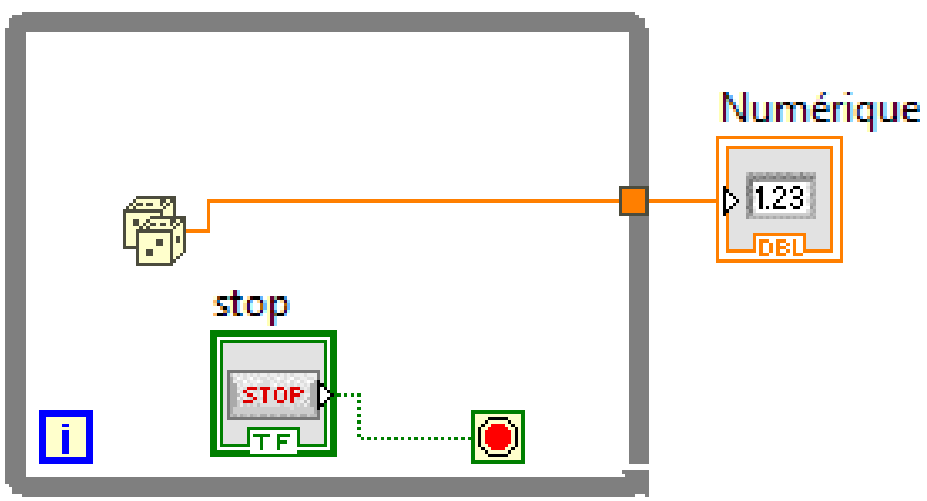
Selon le principe Dataflow, vous devez placer un terminal à l'intérieur de la boucle si vous voulez que ce terminal vérifie ou se mettre à jour à chaque itération de boucle.

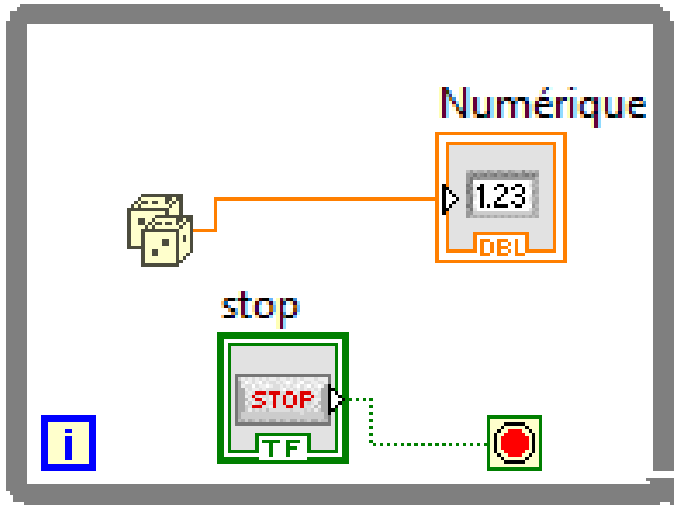


Contrôle de l'exécution des programmes – Tunnel

Dataflow / Tunnel

Selon le principe Dataflow, vous devez placer un terminal à l'intérieur de la boucle si vous voulez que ce terminal vérifie ou se mettre à jour à chaque itération de boucle.

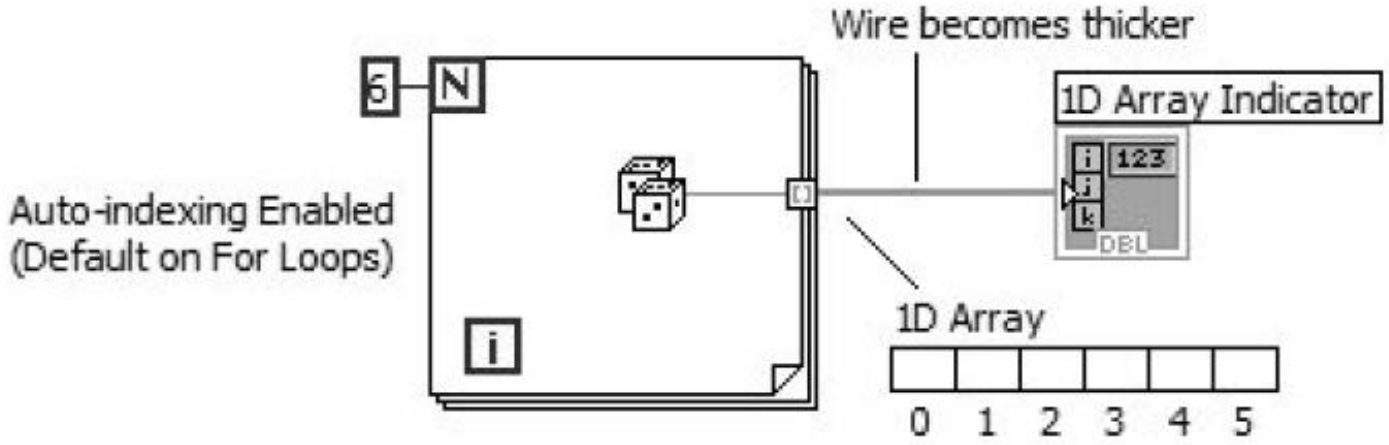




Contrôle de l'exécution des programmes – Auto-Indexing

Auto-Indexing

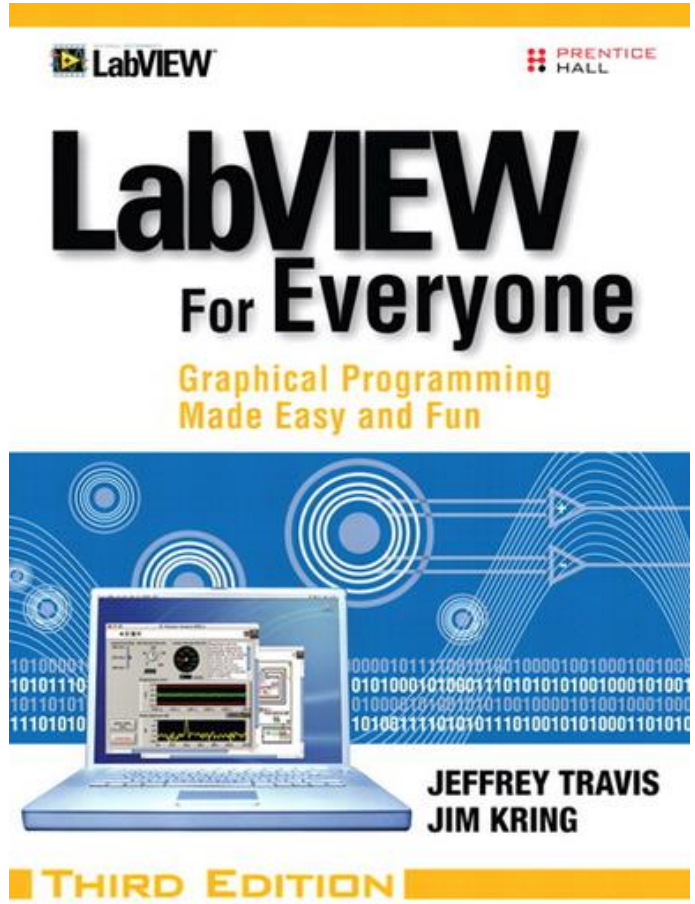
Un boucle While ou For ont la capacité d'emmagasiner les valeurs obtenues à chaque itération et de les insérer dans une tableau. Cette fonction s'appelle auto-indexing. Si vous ne vous pas utiliser cette fonctionne, enlever l'option comme illustré ci-dessous.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

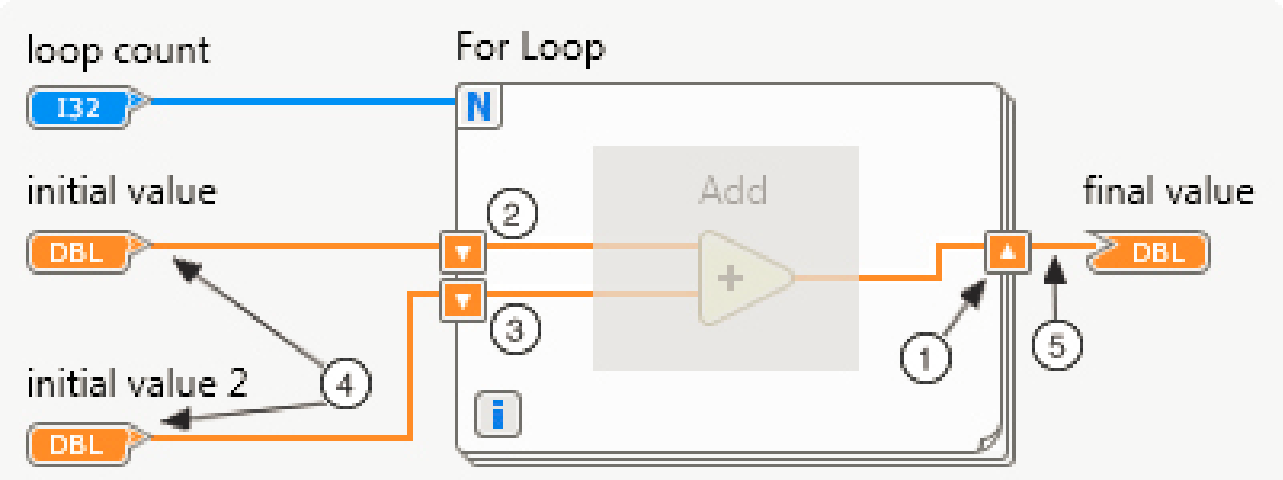
Veuillez vous rendre à la page 190 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-1.



L'environnement LabVIEW – Pratique

Registre à décalage

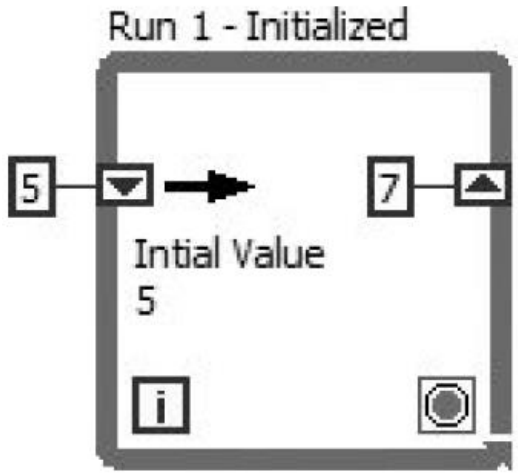
Le registre à décalage est utiliser pour transférer des valeurs du sous-diagramme de la boucle d'une itération à la boucle suivante.



L’environnement LabVIEW – Registre

Initialisation / Registre à décalage

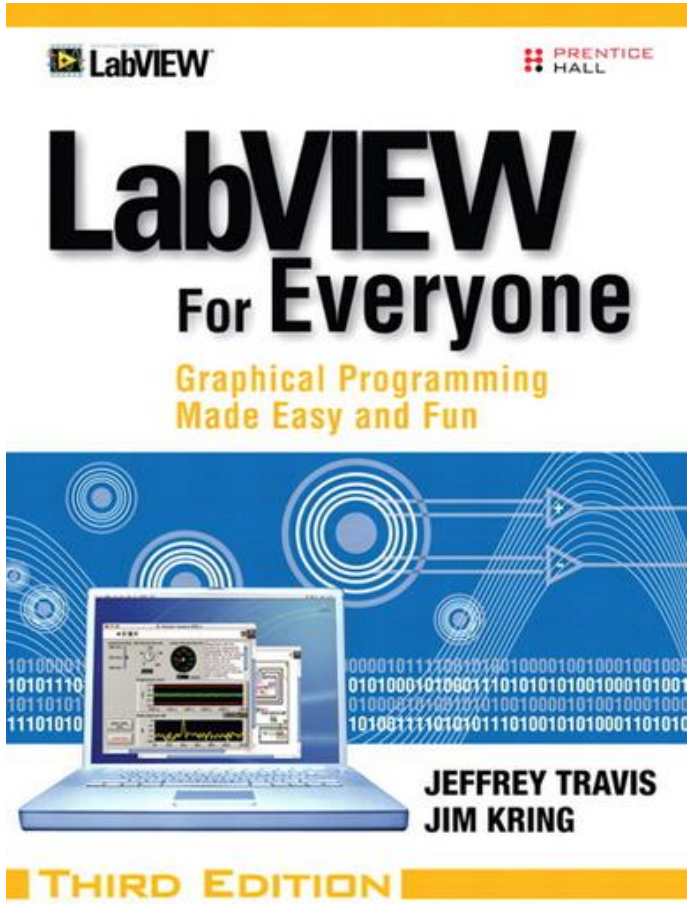
Le registre à décalage est utiliser pour transférer des valeurs du sous-diagramme de la boucle d’une itération à la boucle suivante.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

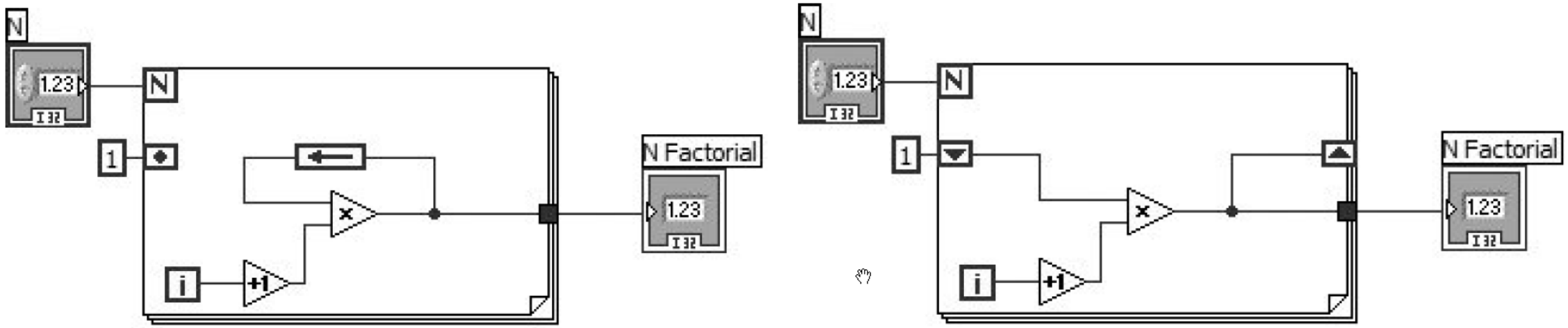
Veuillez vous rendre à la page 197 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-2.



L'environnement LabVIEW – Registre

Feedback Node

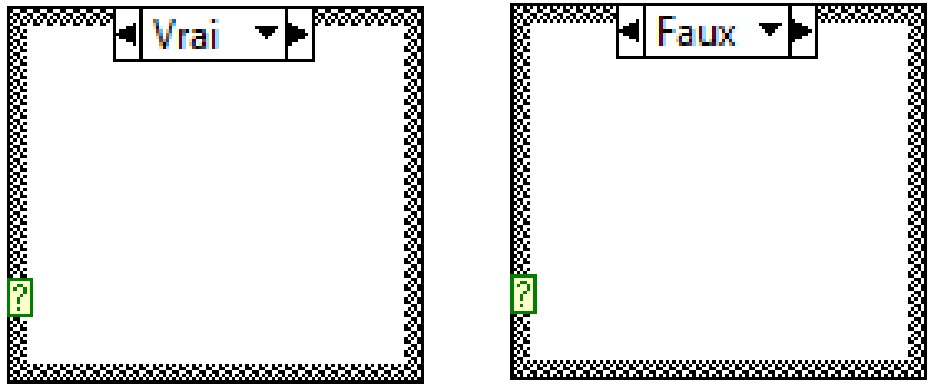
Le Feedback Node permet d'utiliser une valeur à la sortie d'une fonction à l'entrée de cette même fonction pour l'itération suivante. Le Feedback Node est seulement un Shift Register déguisé.

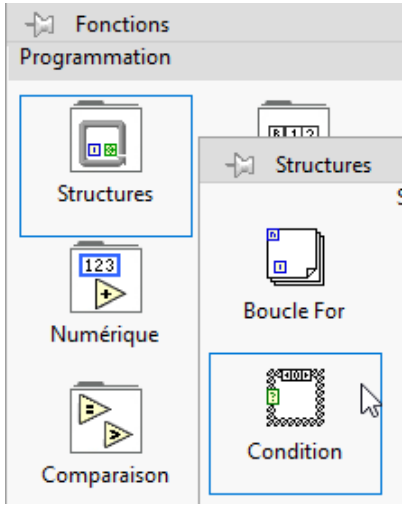


L'environnement LabVIEW – Structure Case

Structure Case

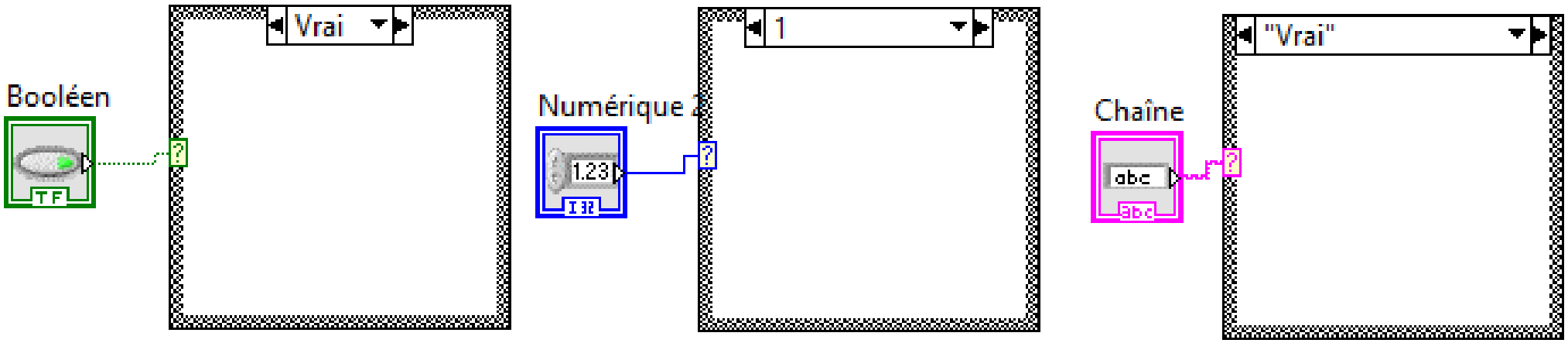
Une structure de type *Case* est une méthode pour exécuter du code selon une condition, semblable à du « If-Then-Else ».





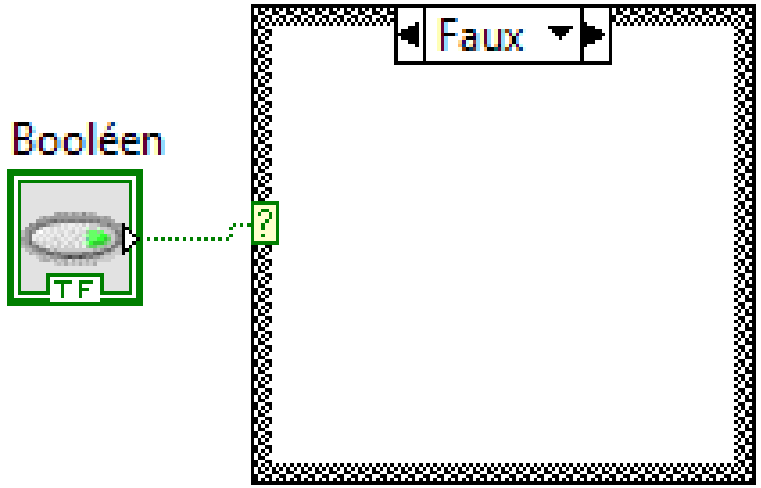
L'environnement LabVIEW – Structure Case

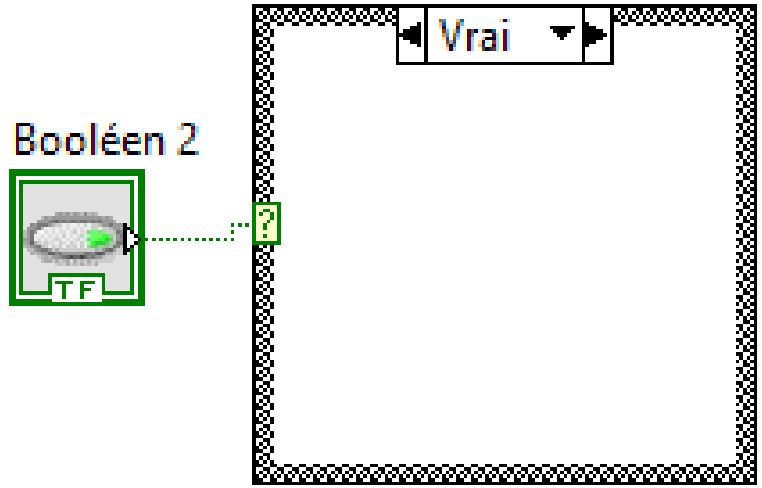
La structure Case a deux ou plusieurs sous-diagramme, ou cases et seulement un de ceux là s'exécute selon la valeur câblée au *Terminal de Sélection.*



L'environnement LabVIEW – Structure Case

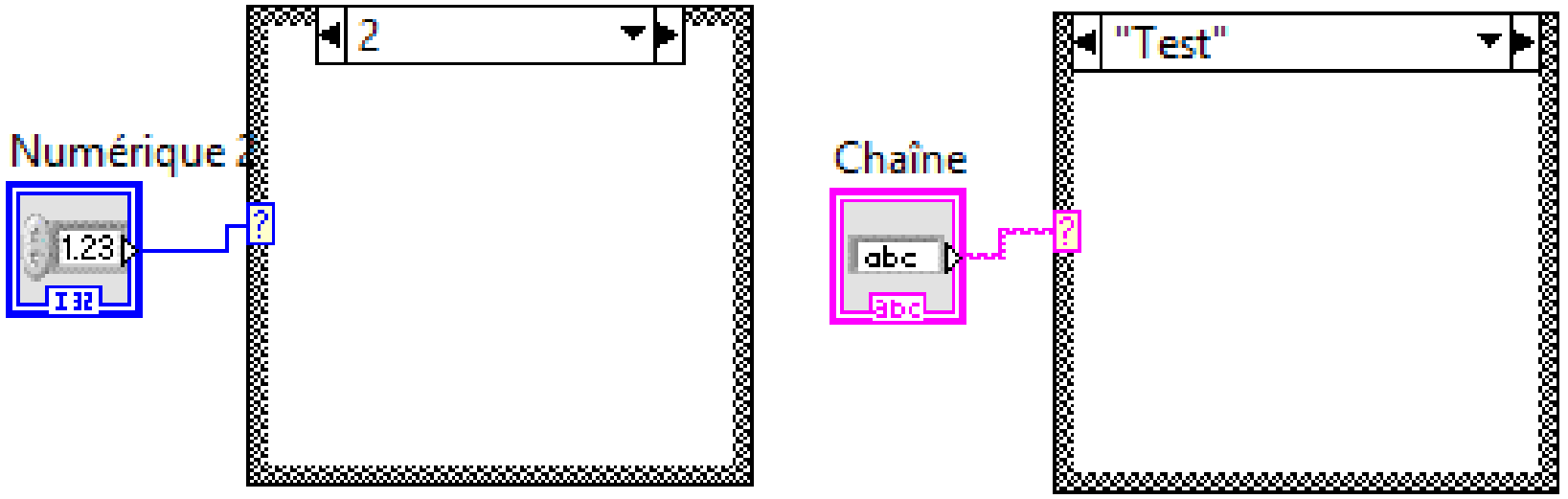
Si une valeur Booléen est câblée au Terminal de sélection, la structure à deux sous-diagramme, VRAI ou FAUX.





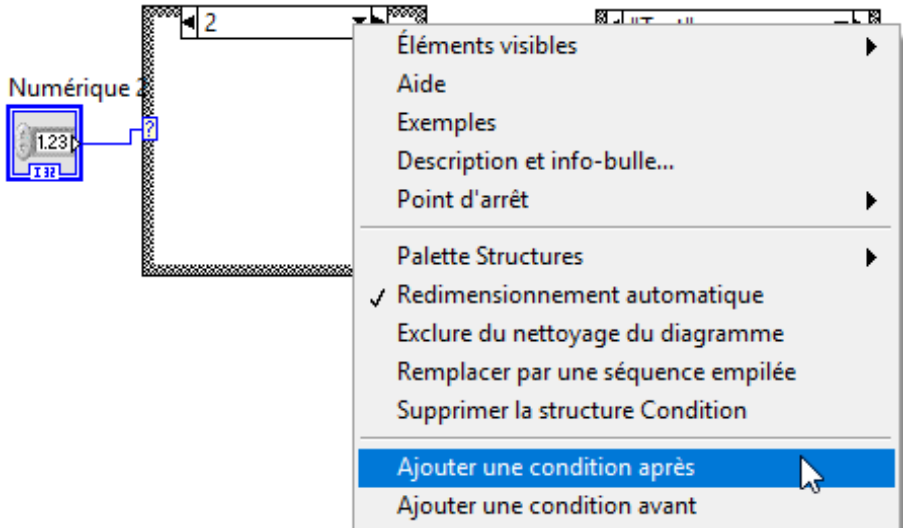
L'environnement LabVIEW – Structure Case

Si une valeur *numérique* ou *string* est câblée au Terminal de sélection, la structure peut avoir de une à un nombre infini de sous-diagramme.



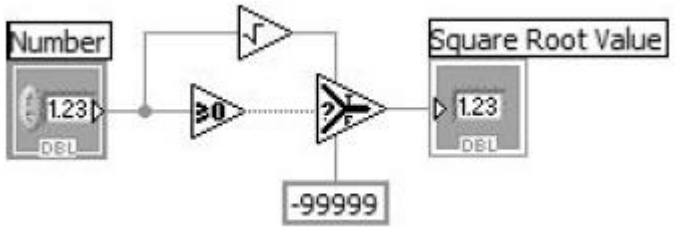
L'environnement LabVIEW – Structure Case

Si vous désirez ajouter une condition, vous pouvez le faire avec le menu contextuel comme voici :



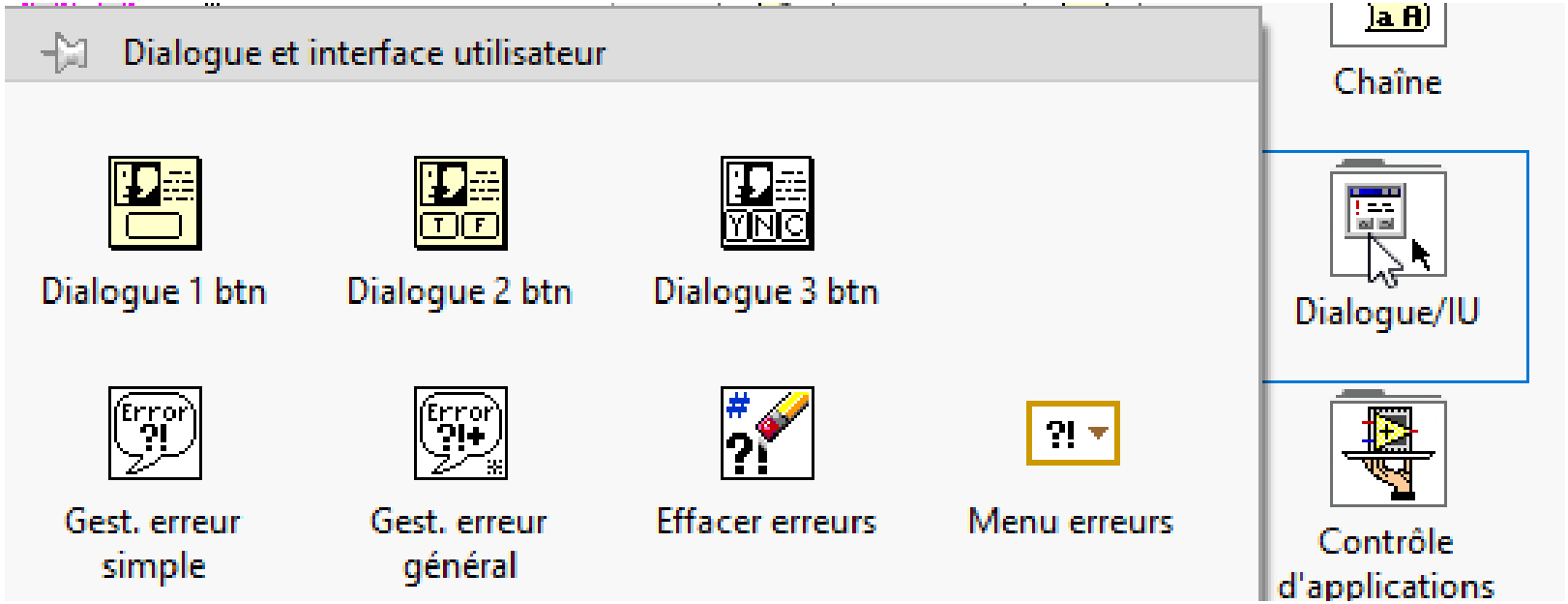
L’environnement LabVIEW – Fonction Select

Dans le cas ou vous devez travailler avec deux options de sortie comme dans le cas d’un simple "if-then-else" , il peut être utile d’utiliser la function Select que l’on retrouve dans les fonctions de base.



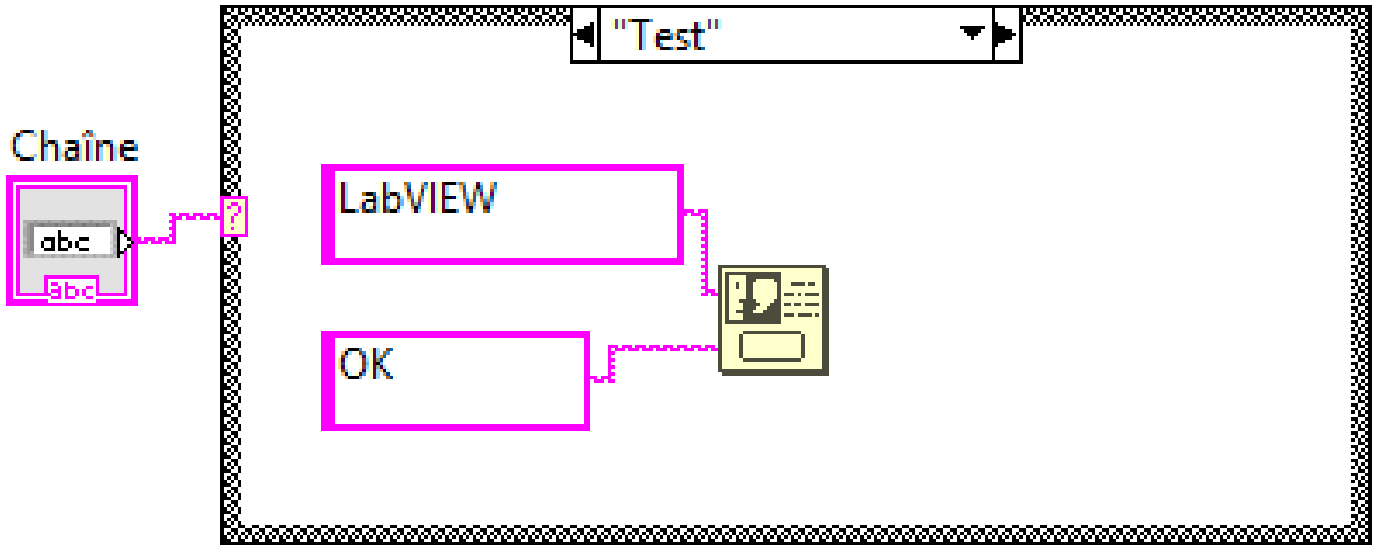
L'environnement LabVIEW – Fenêtres de dialogue

Vous pouvez utiliser la fonction Dialogue afin de faire afficher une fenêtre de dialogue.



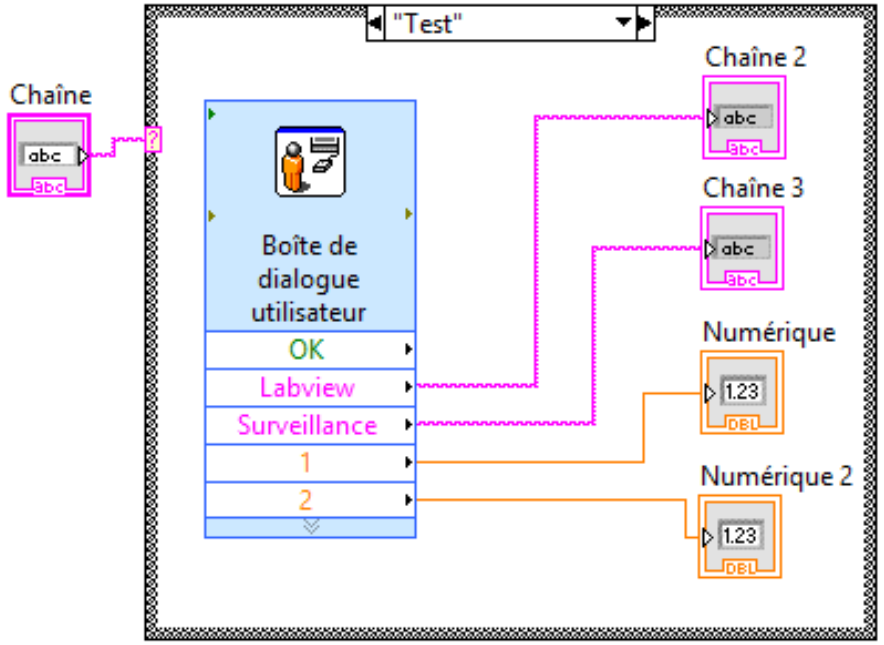
L'environnement LabVIEW – Fenêtres de dialogue

Exemple : Si la condition est respectée, la fenêtre de dialogue s'affichera.



L'environnement LabVIEW – Fenêtres de dialogue

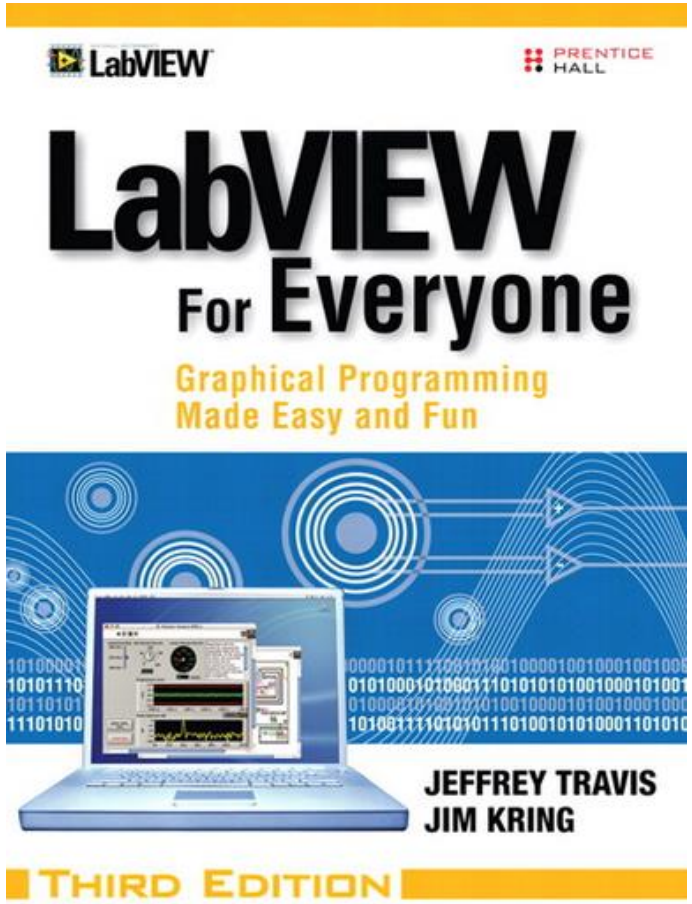
Exemple : Si la condition est respectée, la fenêtre de dialogue s'affichera.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

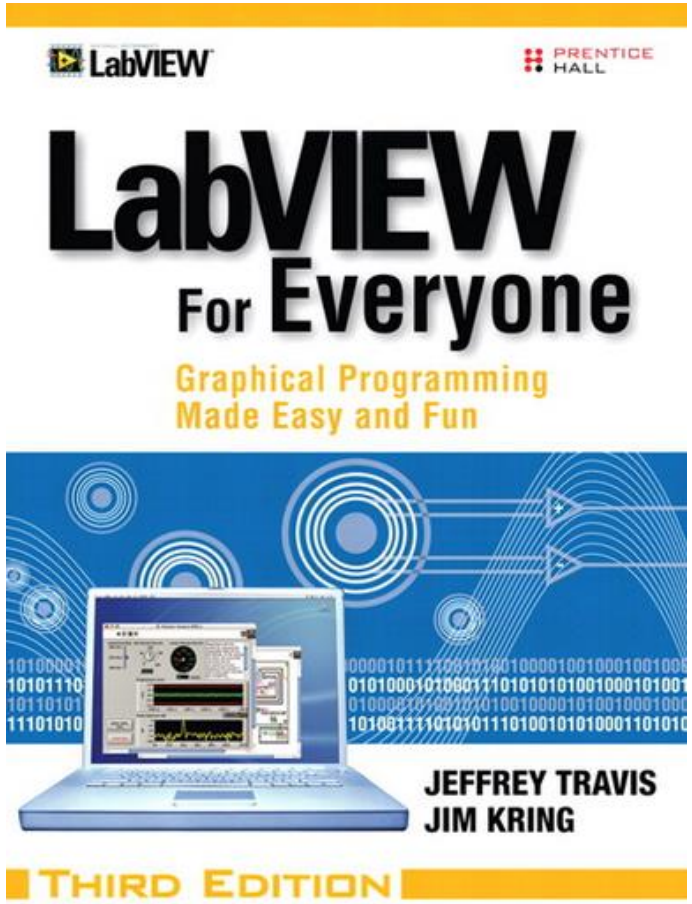
Veuillez vous rendre à la page 213 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-3.



L'environnement LabVIEW – Pratique

Après avoir lu les pages 209 à 212, créez un VI contenant un invite de commande demandant pour un nom d'utilisateur et un mot de passe.

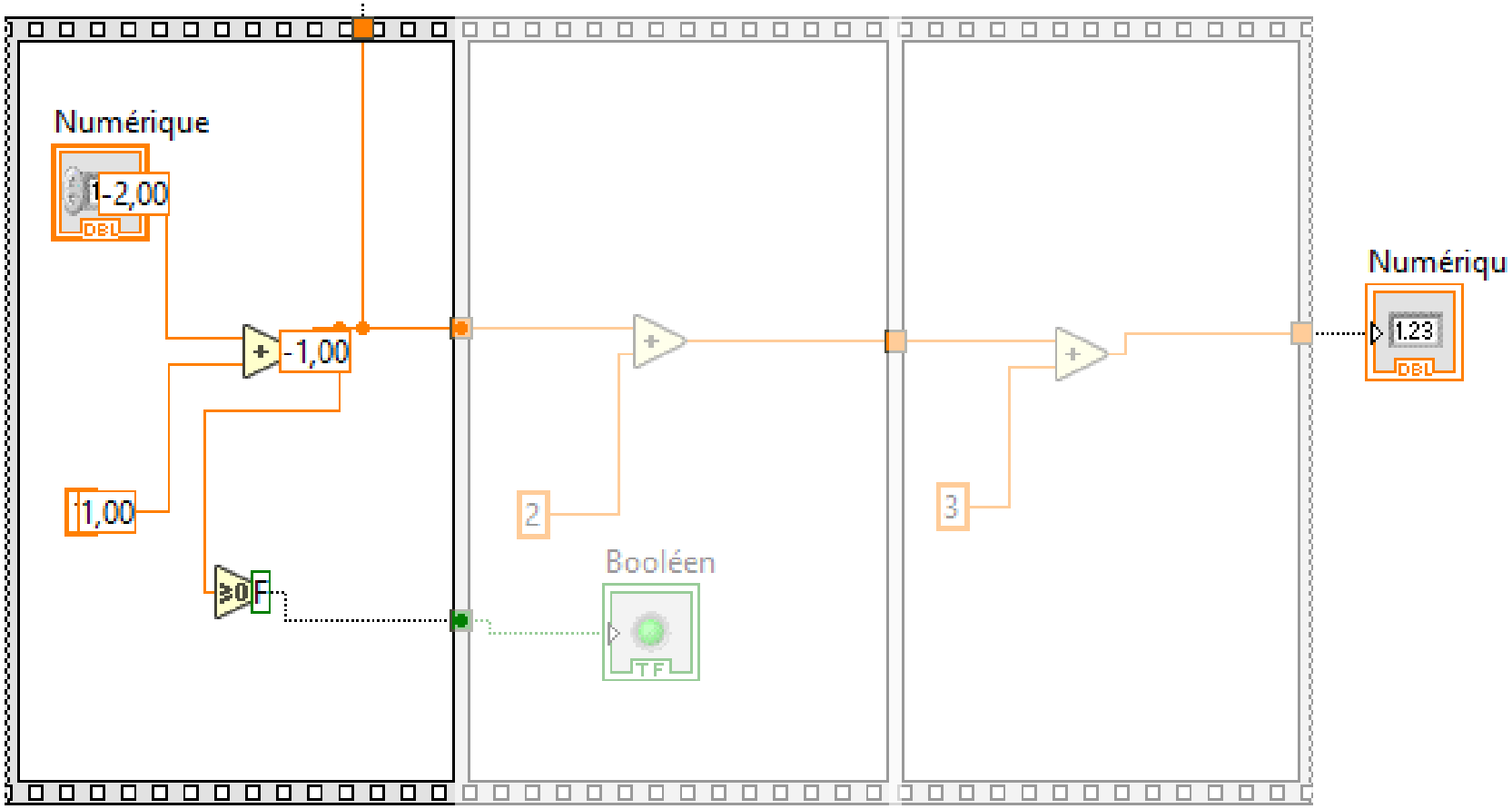
Par la suite, amusez-vous à concevoir une VI avec plusieurs types de messages et dialogues.



L'environnement LabVIEW – Sequence structure

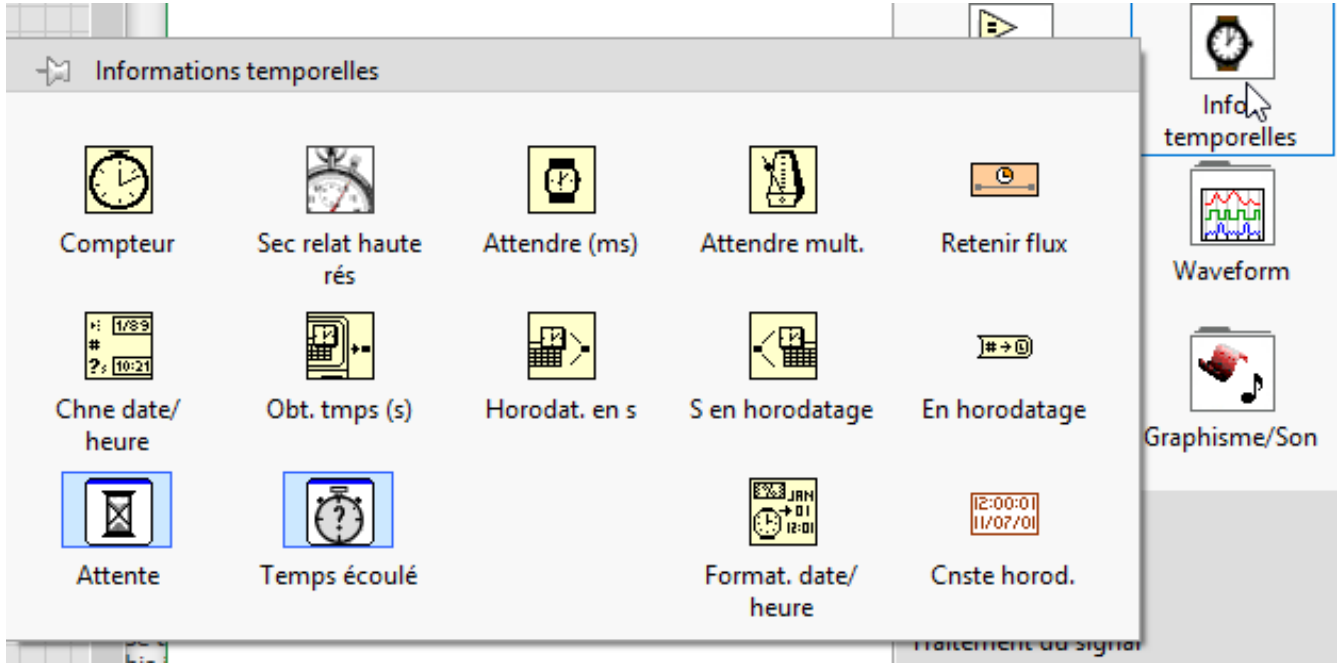
Une structure de séquence est un ensemble de « frames » qui s'exécute de façon séquentielle. Elle exécute le *« Frame 0 »* suivi du *« Frame 1 »*. Les données quittent la structure seulement lorsque le dernier *« Frame »* est complété et que toutes les données de la séquence arrivent à la limite de celle-ci.

L'environnement LabVIEW – Sequence structure



L'environnement LabVIEW – Chronométrage

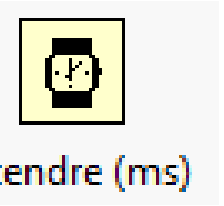
Les fonctions de chronométrage sont très importante dans LabVIEW. Elles permettent de mesurer le temps, synchroniser les tâches et de permettent un temps d'inactivité au processeur.

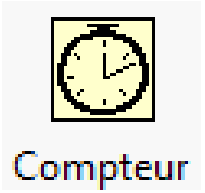


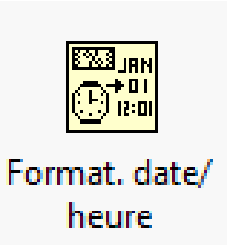
L'environnement LabVIEW – Chronométrage

Il y a plusieurs fonctions de chronométrage et vous pouvez vous en servir pour différents applications.

Exemple : Attendre X (ms) à chaque itération de boucle.



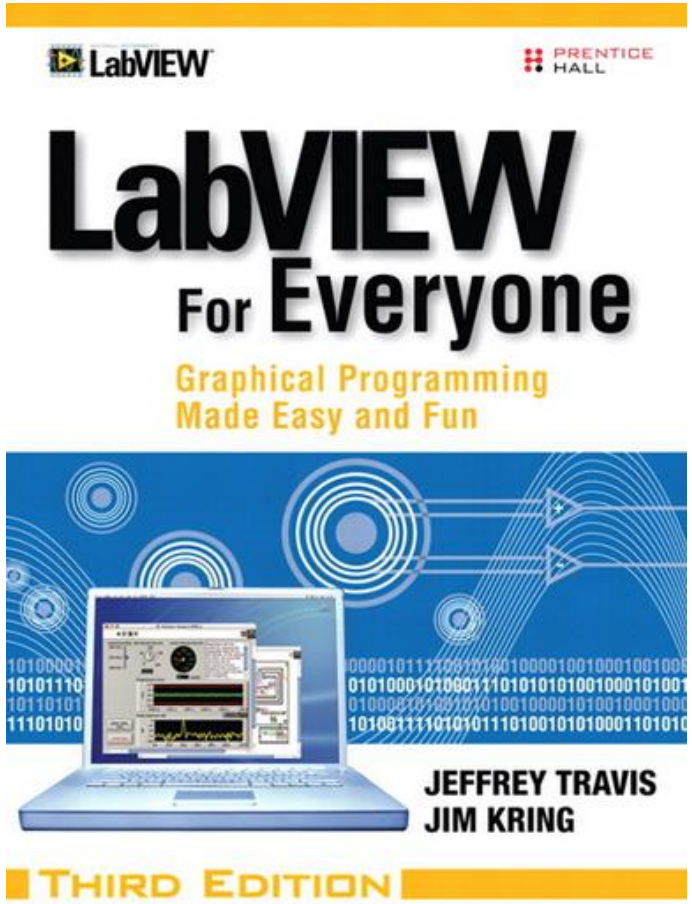




L'environnement LabVIEW – Pratique

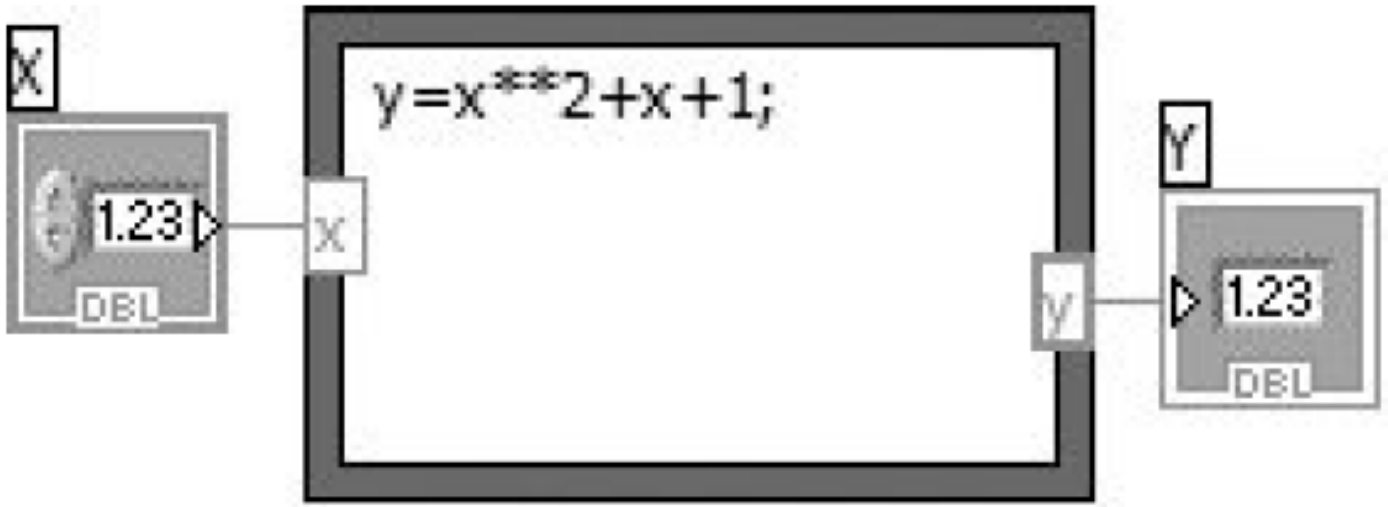
C'est le temps de passer à la pratique!

Veuillez vous rendre à la page 221 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-4.



L'environnement LabVIEW – Formula node

Le Formula Node est une boîte où vous pouvez entrer des formules directement dans le Block Diagram avec plusieurs entrées et sorties disponibles..

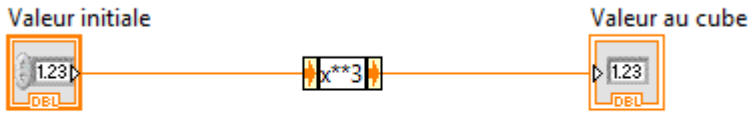


L'environnement LabVIEW – Expression node

Le Expression Node est une boîte où vous pouvez entrer des formules directement dans le Block Diagram mais seulement avec une entrée et une sortie possible.

Contrairement au **Formula Node**, nous n'avons pas à entrer le nom des entrées et sorties (comme il n'y a qu'une entrée et une sortie, il ne peut y avoir d'ambiguité)

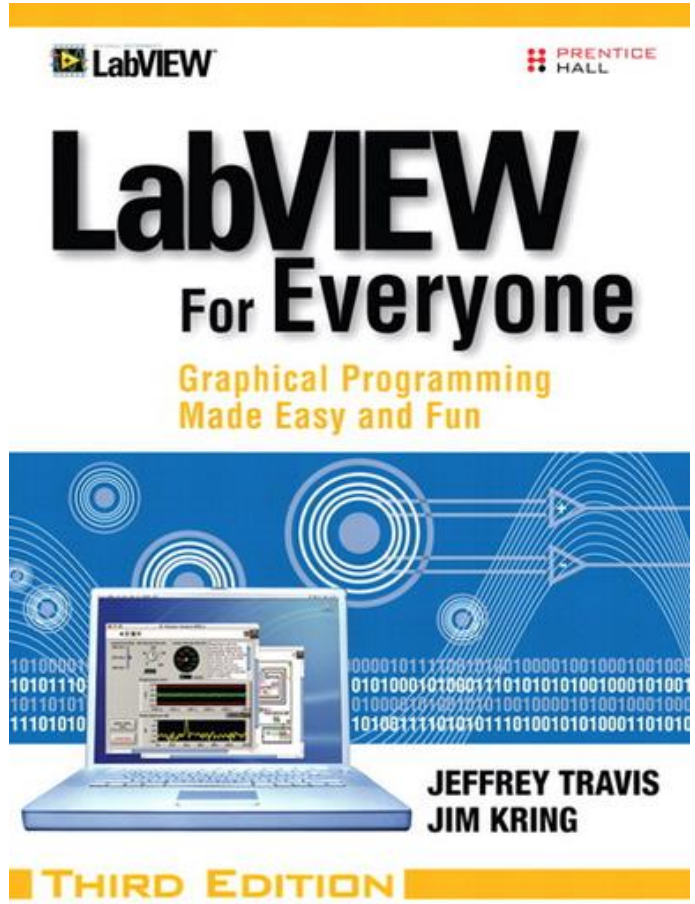
De plus, pas besoin de mettre de point virgule à la fin de l'expression mathématique.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

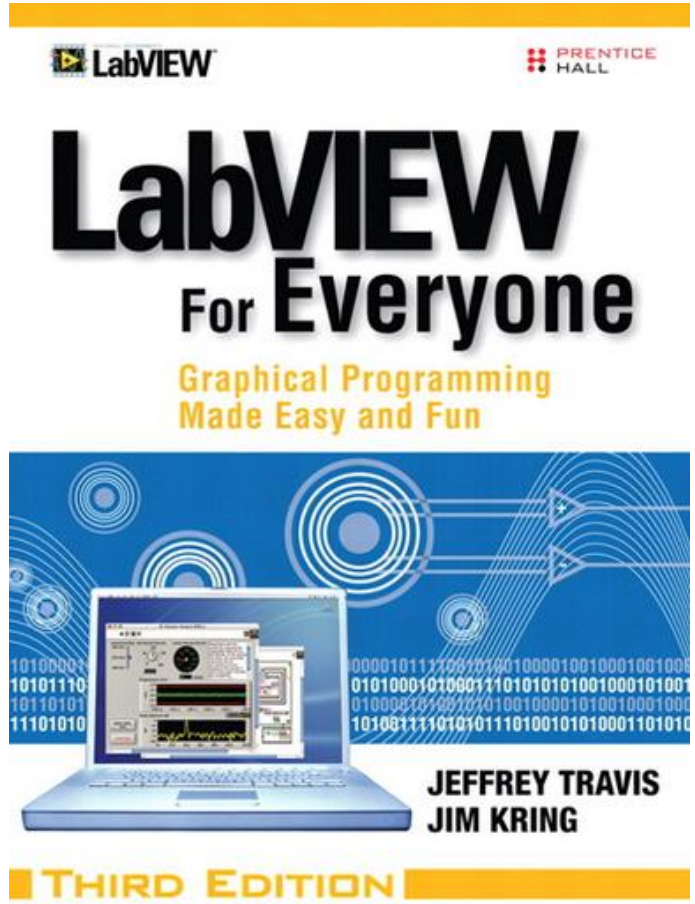
Veuillez vous rendre à la page 231 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-5 et 6-6.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

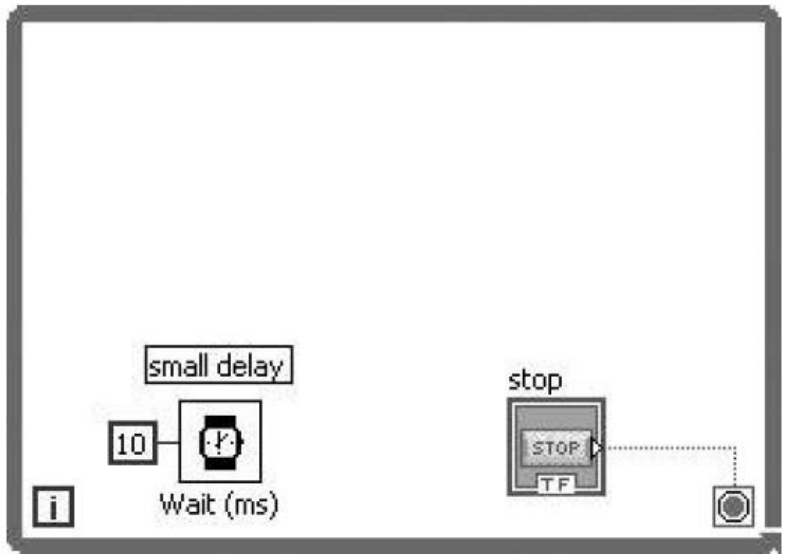
À l'aide de formula node et des toutes les notions vues jusqu'à présent, veuillez créer un VI qui permet de visualiser une équation du deuxième ordre. Nous devons être en mesure de modifier les paramètres de l'équation Ax^2+Bx+C. Vous devrez aussi trouver la façon d'utiliser la fonction graphique XY.



L'environnement LabVIEW – Boucle While + Case Structure

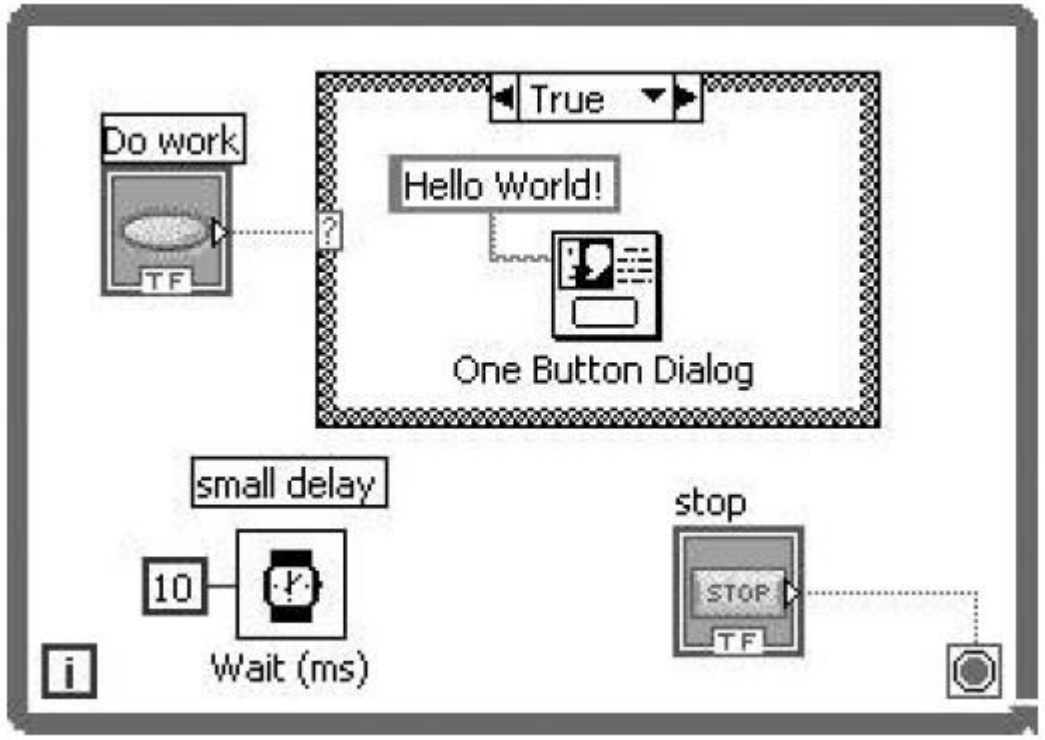
La boucle principale / Main Loop

Toutes les applications écrites avec LabVIEW devrait avoir au moins une boucle principale afin de terminer le programme avec un bouton STOP. Ajouter un petit délai l'intérieur de cette boucle permet de ne pas encombrer le processeur de l'ordinateur.



L'environnement LabVIEW – Boucle While + Case Structure

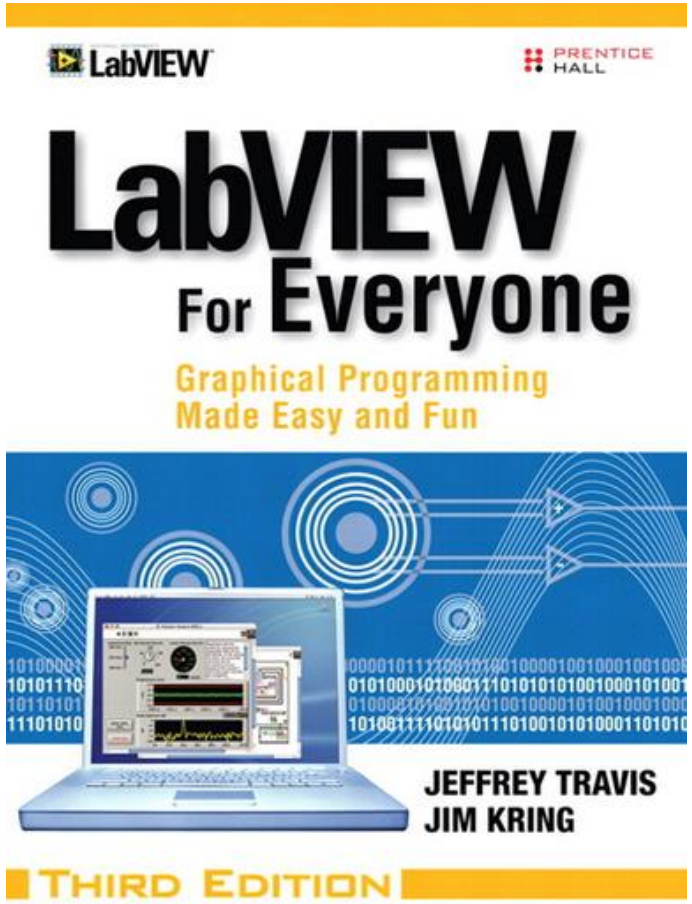
Une boucle While + une Structure Case est la clé pour créer des applications logiciels avec Labview.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

Veuillez vous rendre à la page 241 de votre livre *LabVIEW for Everyone* et compléter l'activité 6-7. Vous devez aussi intégrer une Main Loop à votre programme avec bouton d'arrêt.



MODULE #1

243-575-RK (3-2-3)

PARTIE 4

STRUCTURES DE DONNÉES

Enseignant : Sébastien Richard

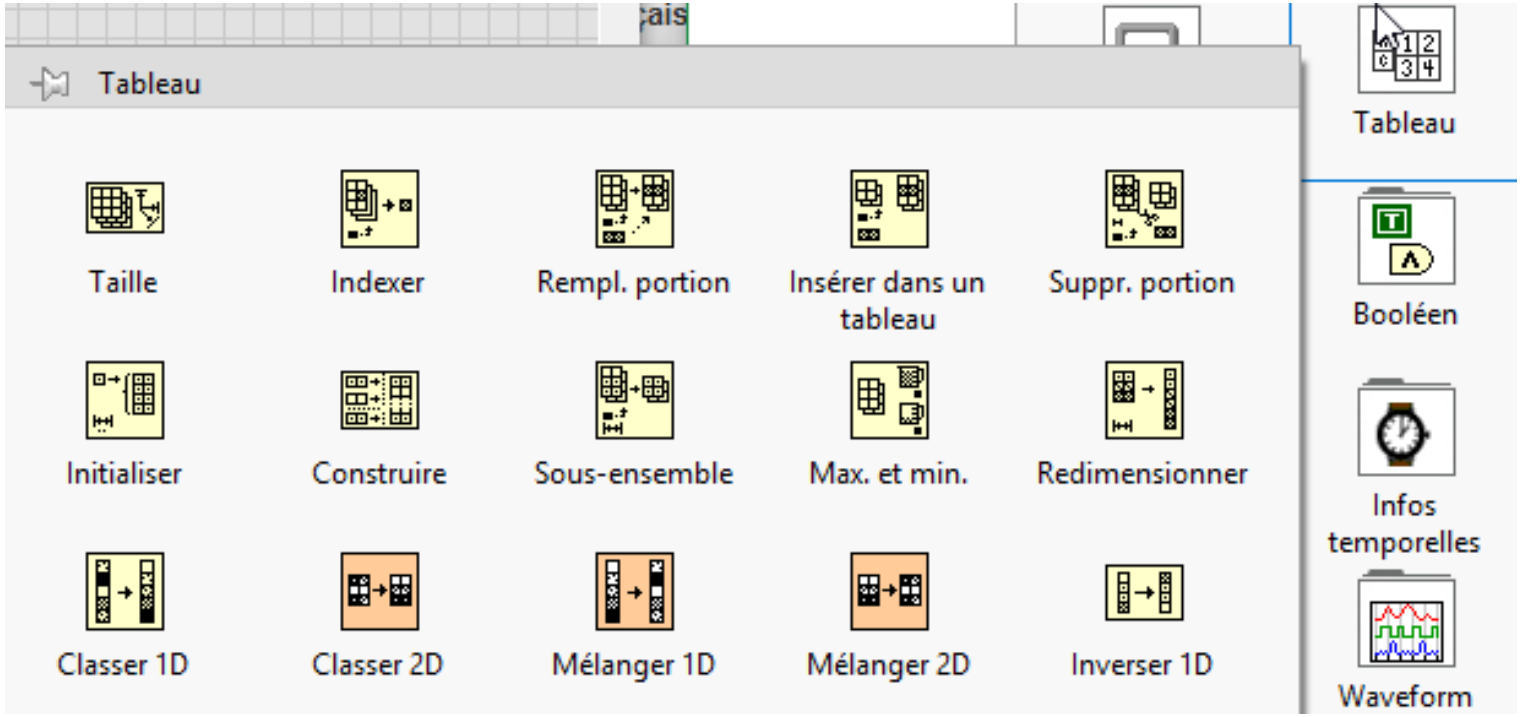
Structures de données LabVIEW

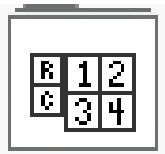
- Apprendre à utiliser les fonctions de manipulation d'*Arrays*
- Apprendre à utiliser les *Clusters*
- Apprendre à différencier les *Clusters* des *Arrays*

Structures de données LabVIEW

Arrays / Tableaux

Un Array / Tableau est une collection d'éléments de données qui sont tous du même type. (Booléen / Numérique / String).



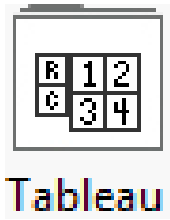


Structures de données LabVIEW

Arrays / Tableaux

Les éléments du tableau sont accessible par leurs indices. Chaque index des éléments est dans la gamme 0 à N-1 où N est le nombre total d'éléments dans le tableau.

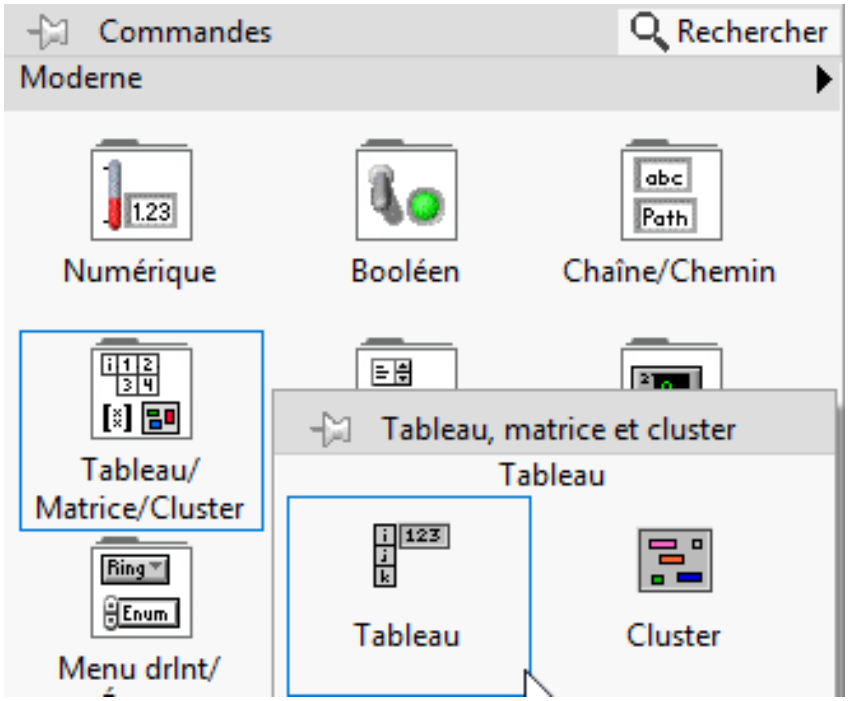
Index	0	1	2	3	4	5	6	7	8	9
10-Element Array	12	32	82	8.0	4.8	5.1	6.0	1.0	2.5	1.7

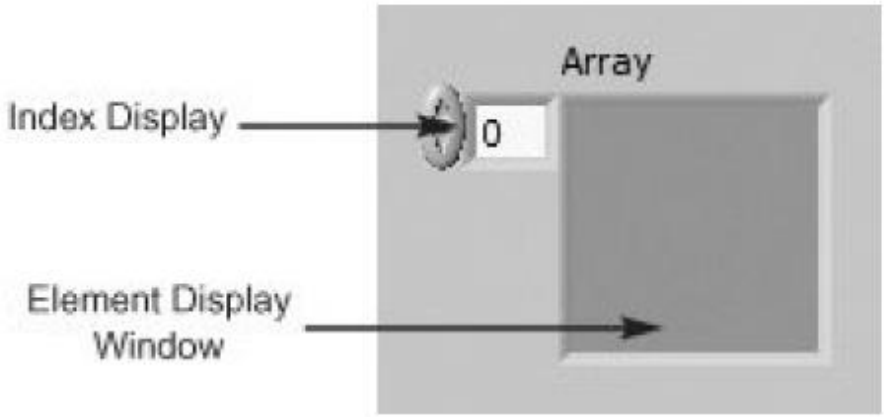


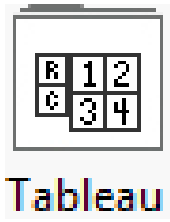
Structures de données LabVIEW

Création des Arrays

Pour créer un Array, il faut combiner un Array Shell (Coquille) avec un objet de données dans le Front Panel.



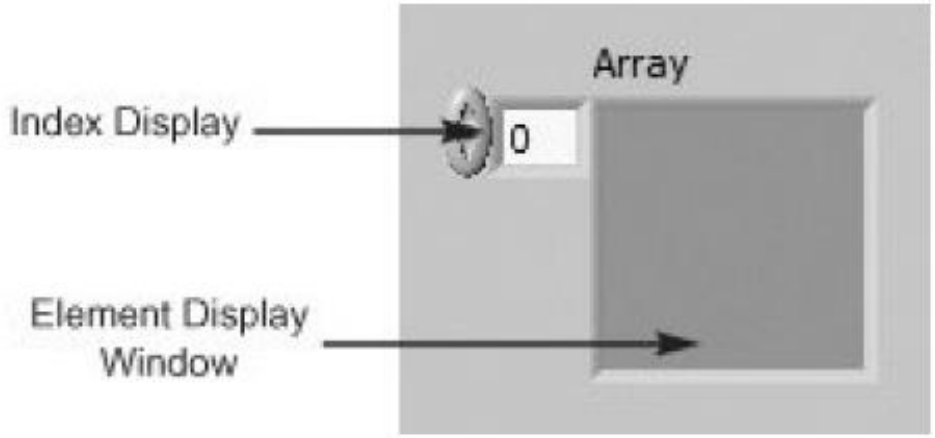


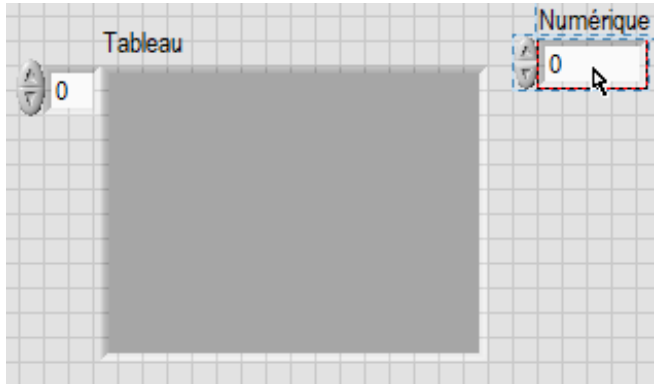


Structures de données LabVIEW

Création des Arrays

Lorsque la *coquille/Array Shell* est en place, il s'agit de faire glisser l'objet de données à l'intérieur de la *Fenêtre d'affichage d'élément*.



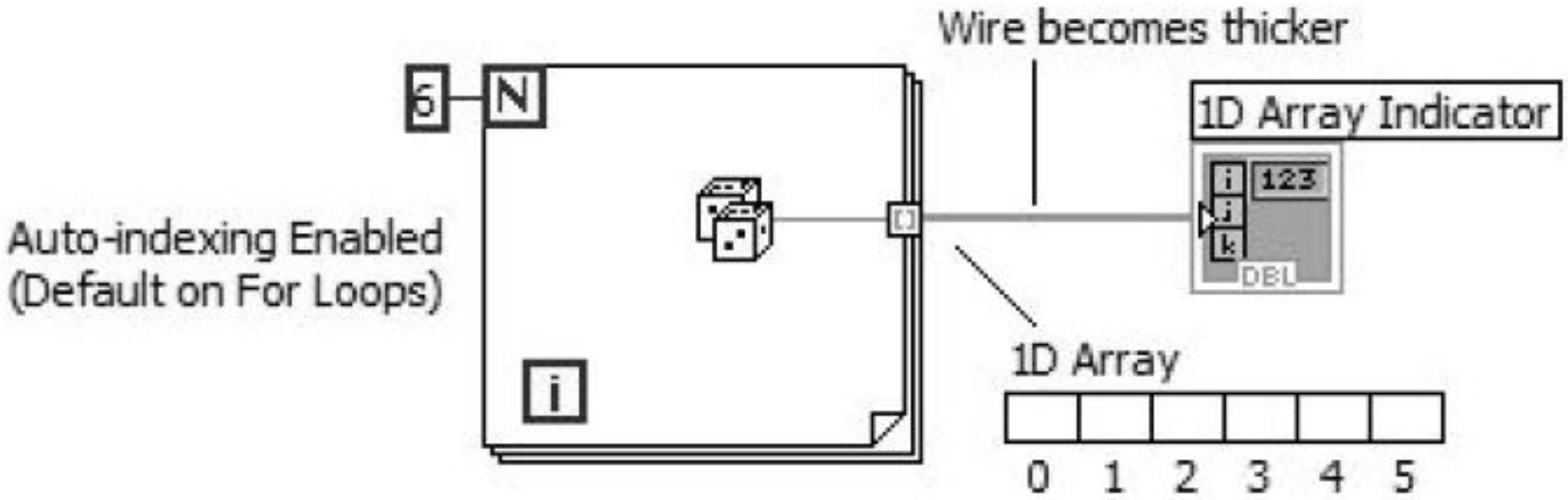


Structures de données LabVIEW

Tableau

Auto-Indexation

Les boucles For et While peuvent indexer et accumuler les données, les éléments automatiquement à leurs bordures.



L'environnement LabVIEW – Pratique

C'est le temps de passer à la pratique!

1. Vous devrez créer un programme qui permet de visualiser soit un signal sinusoïdal ou cosinusoïdal selon le choix de l'utilisateur. L'usager doit pouvoir modifier l'amplitude, la fréquence (en Hz) et le déphasage du signal (en degré) demandé. Vous devez aussi afficher à l'écran la période. Votre graphique devra afficher ce signal pour des fréquence de 10 à 100 Hz. Vous devez limiter l'utilisateur pour ne pas aller au-delà de ces valeurs. L'étendu de votre graphique devra être d'une durée de 100ms et devra compter au moins 100 données.
2. De plus, il faut être en mesure de visualiser les 100 données de façon numérique à l'aide d'indicateurs sur votre interface usager sous forme de deux colonnes et aussi de deux rangées.
3. Vous devez aussi afficher le résultat dans un fichier excel sous forme de deux colonnes et de deux rangées

MODULE #1

243-575-RK (3-2-3)

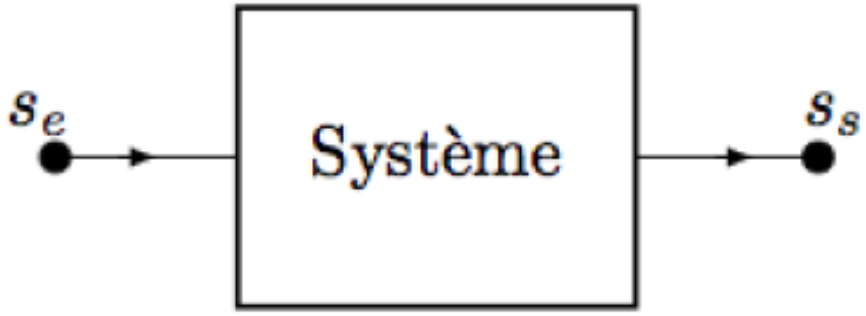
PARTIE 5

LES FONCTIONS DE TRANSFERT

Enseignant : Sébastien Richard

Les fonctions de transfert

Une fonction de transfert est un modèle mathématique de la relation entre l'entrée s_e et la sortie s_s d'un système linéaire. Elle est utilisée, notamment, en traitement du signal.



Pierrick Lotton et Manuel MELON

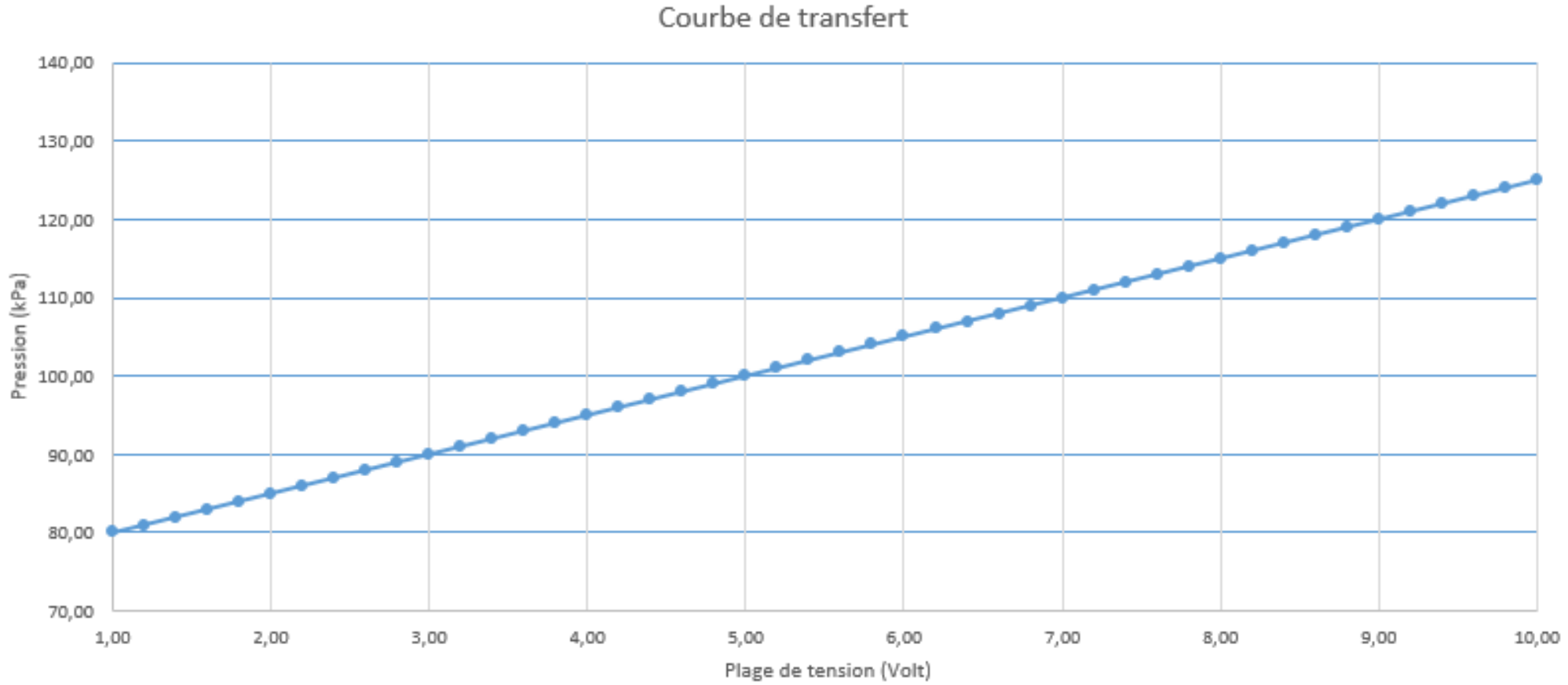
Les fonctions de transfert

Démonstration 1:

Un capteur de température fonctionne dans la plage allant de 10°C à 125°C. Lorsque la température est de 10°C, la tension à la sortie du capteur est de 1V. Elle est de 4V lorsque la température est de 150°C. La courbe de transfert est entièrement linéaire. Donnez la fonction de transfert de ce capteur.

Les fonctions de transfert

Exercice #1: Un capteur de pression fonctionne selon la courbe suivante. Donnez la fonction de transfert de ce capteur.



Les fonctions de transfert

Exercice #2: Un capteur de température fonctionne selon les spécifications suivantes :

- À -50°C, la tension à la sortie du capteur sera de 1 volt
- À 150°C, le tension est de 3 volt.

La courbe de transfert est linéaire, trouvez la fonction de transfert associée.