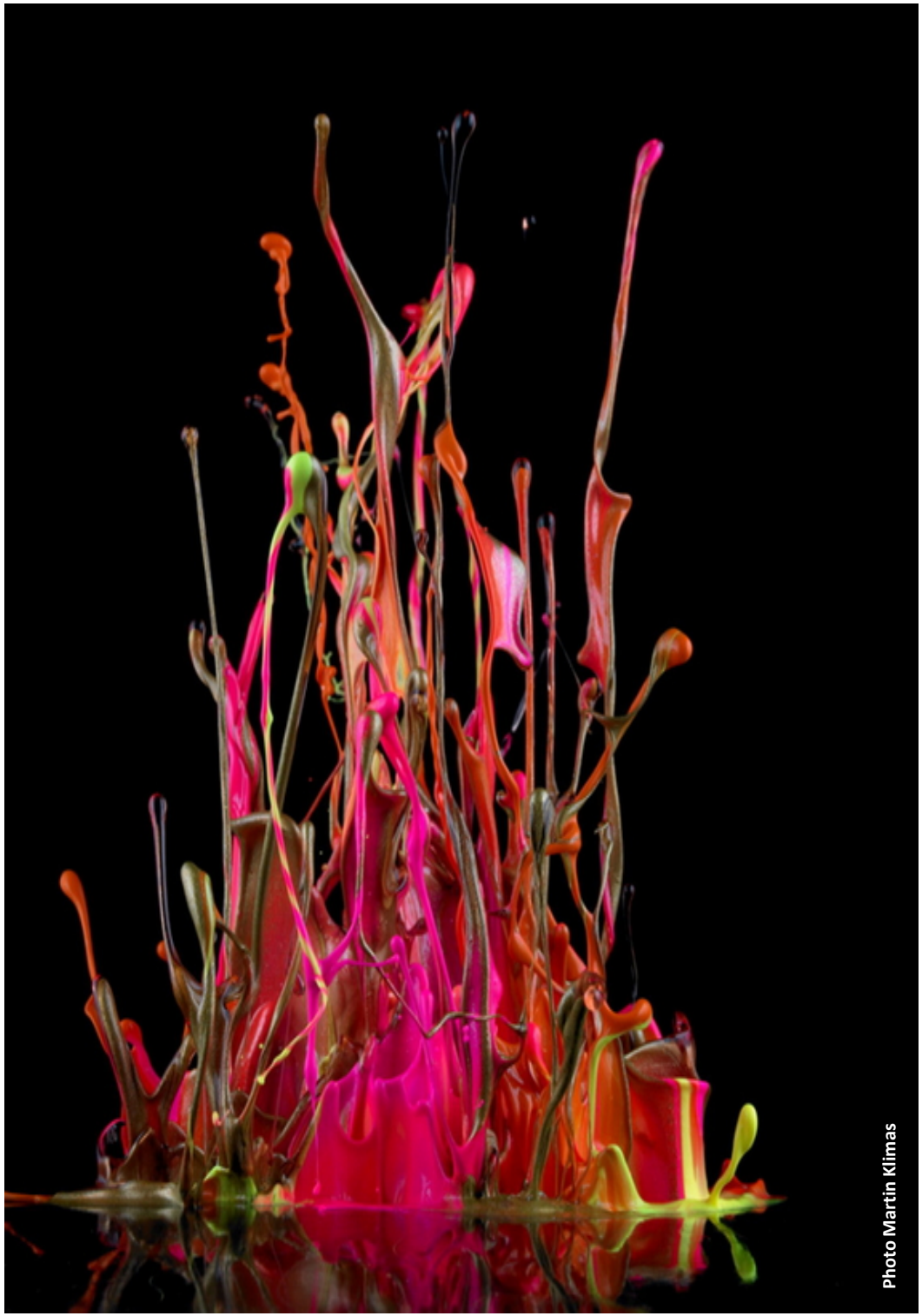


Photo Martin Klimas

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

LabVIEW

Exo - my Third VI

ME 2e semestre

Rev 2018.1

Christophe Salzmann

Laboratoire d'Automatique

My Third VI

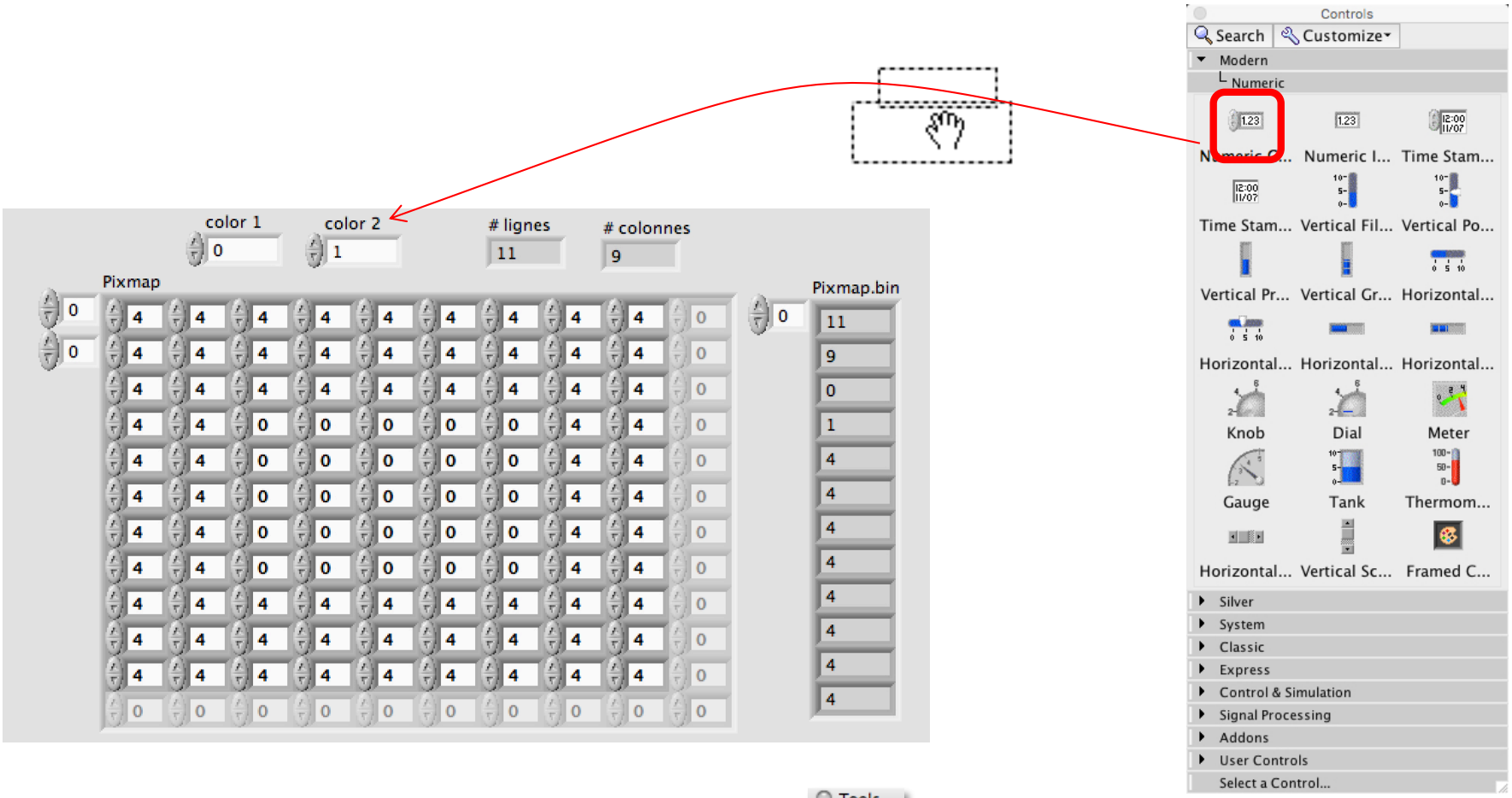
But:

Tester votre programme C++ depuis LabVIEW

Etapes:

- Simuler une image
- Creation du fichier **pixmap.bin**
- Appel de votre code C++
- Lecture du fichier **traces.m**

Dessinez votre interface

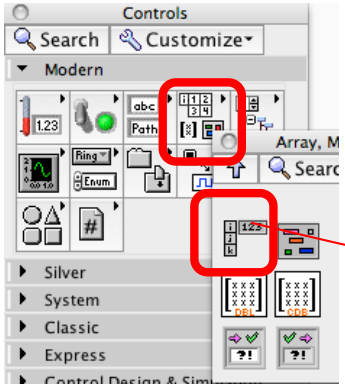


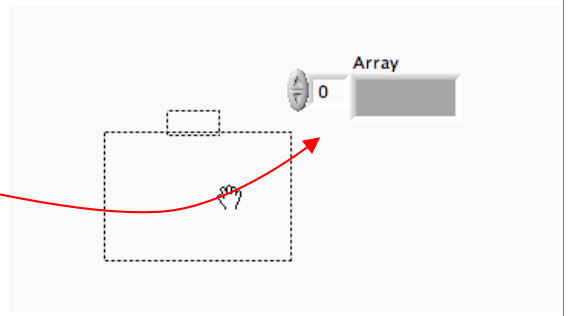
Et éditez le nom de vos contrôles

Array creation

Front Panel

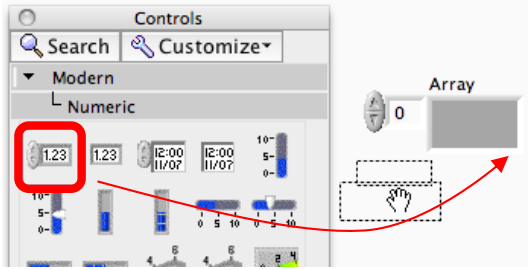
diagram

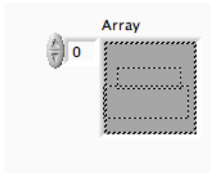




Array

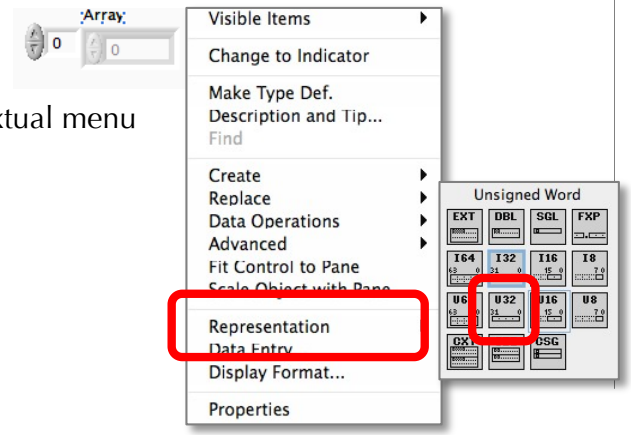
Créer un tableau vide





Array
[DBL]

Glisser un élément pour lui donner un type



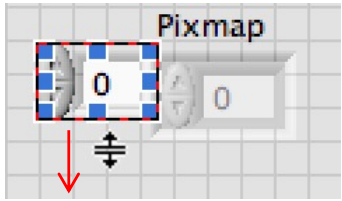
right click for contextual menu

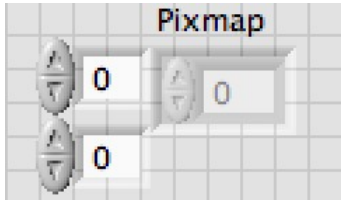
Array
[U32]

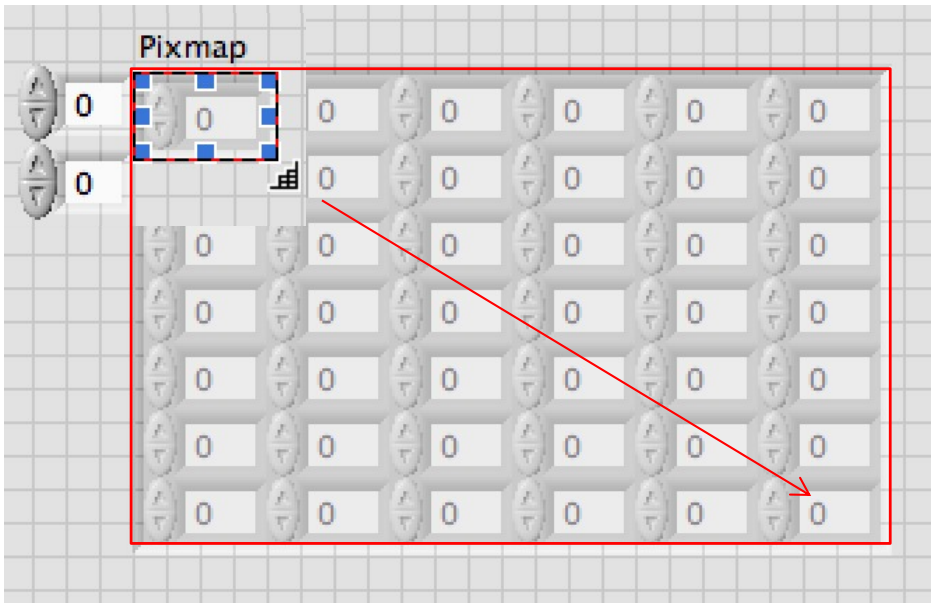
Changer le type des éléments du tableau en UInt 32

Array creation

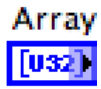
Front Panel



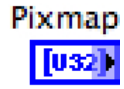




diagram



Ajouter une dimension à votre tableau
Et changer son nom

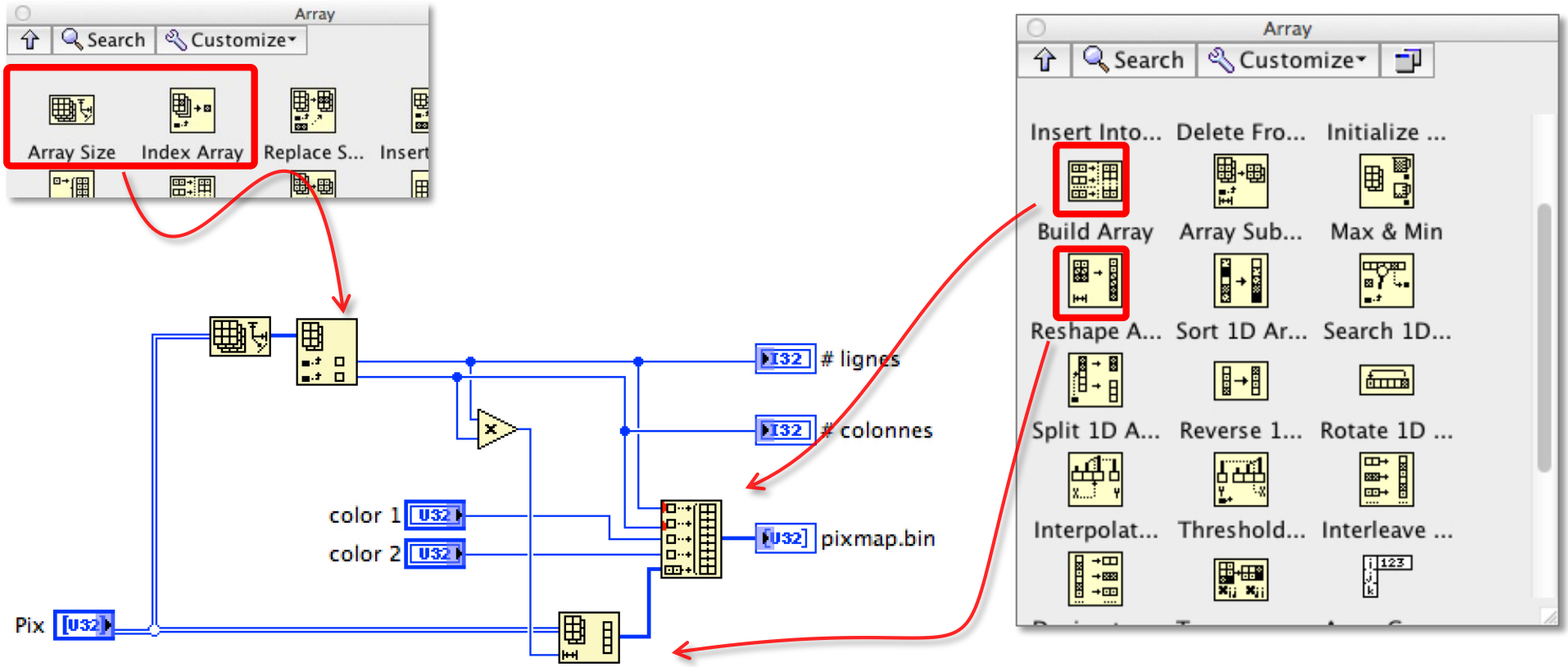


Notez que les [..] sont plus larges

Agrandir le tableau pour voir plusieurs éléments

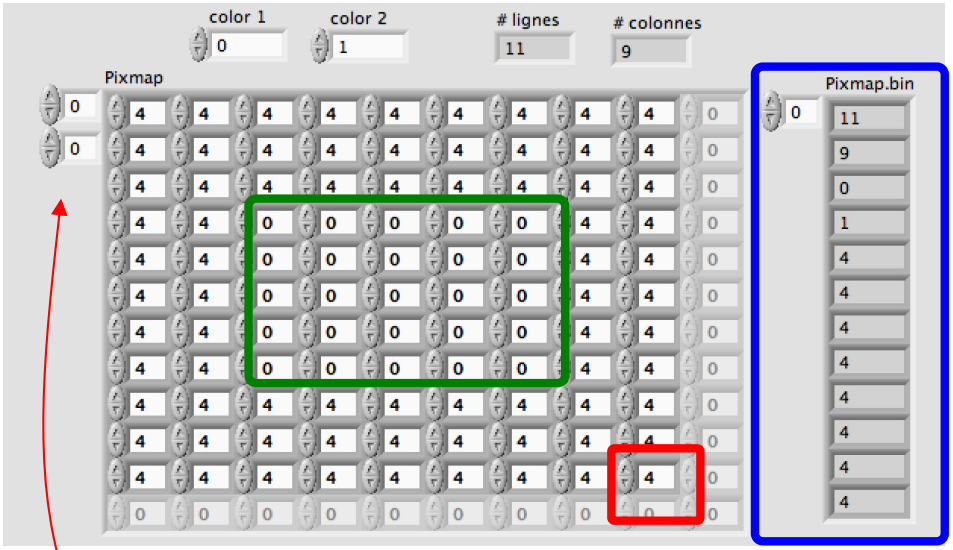
Les cases grisées sont vides

Agréger les *controls* en [U32]



Construisez votre tableau (ici un vecteur) "Pixmap.bin". Le VI **Reshape Array** met les lignes du tableau Pixmap les unes à la suite des autres. Le VI **Insert Into Array** ordonne les differents éléments du de pixmap.bin.

Remplir le tableau

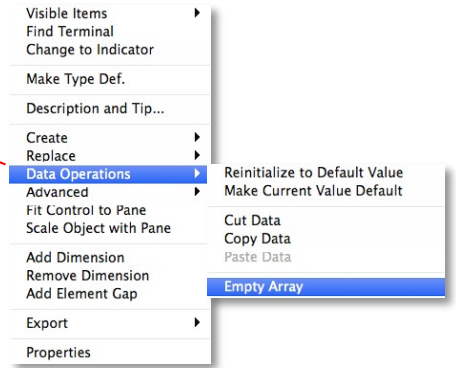


Commencez par cliquer sur l'élément final de votre tableau (en rouge) et lui donner la valeur désirée (ex.4)

Le reste du tableau s'initialise automatiquement à (0)

Remplir les cases de votre choix (vert) avec la valeur des couleurs **color 1 ou 2** pour définir la pattern 5x5 et les autres cases par des couleurs autres (ex. 4)

Executez votre VI et vérifier que votre array 1D **Pixmap.bin** correspond à ce que vous désirez

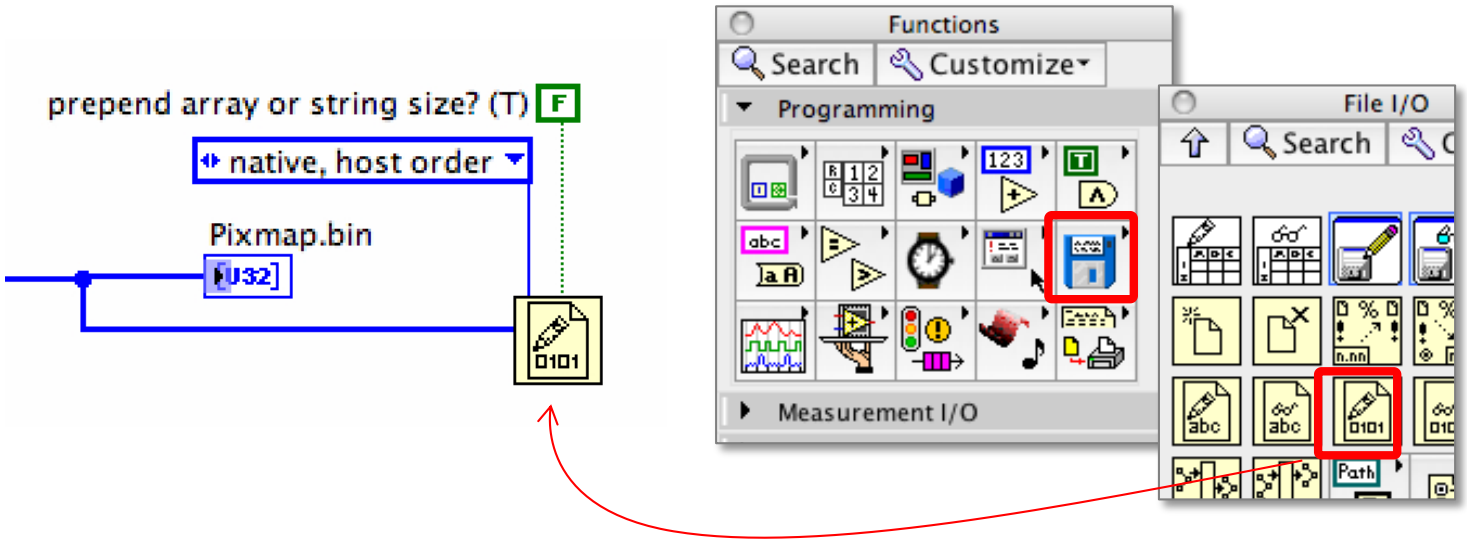


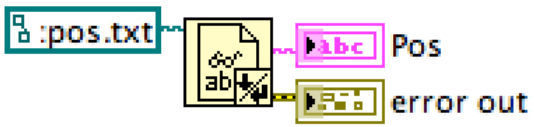
Vous pouvez vider votre tableau en faisant un clic de droite sur les indexes du tableau et en choisissant

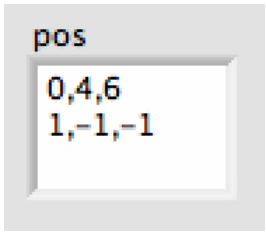
Data Operantion-> Empty Array

Sauvegarde de pixmap.bin

Connectez votre array Pixmap.bin au VI **Write to Binary file**. Afin de ne pas ajouter la taille du tableau au début du fichier mettez **`prepend array or string size? (T)`** à faux (F). De même vous devez ajuster *l'endianess* des données sauvée sur votre disque dur (**`little-endian`** ou **`native, host order`**). Lors de la prochaine exécution de votre VI, vous aurez un dialogue qui vous demande où sauver le fichier. A vous de lui donner le bon chemin ainsi que le nom ***pixmap.bin***







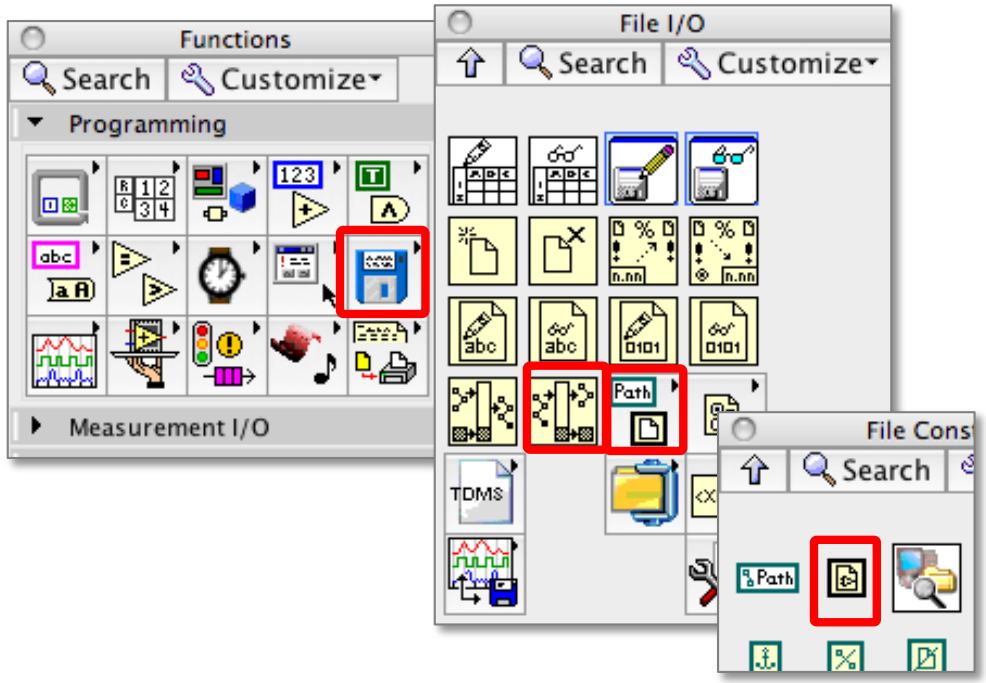
De la même manière vous pouvez lire le fichier **pos.txt** qui sera créé par votre exécutable (slides 10-11)

Chemin courant

La constante **`Current VI's path`** donne le chemin complet sur le VI courant (i.e. le chemin de celui qui contient la constante).

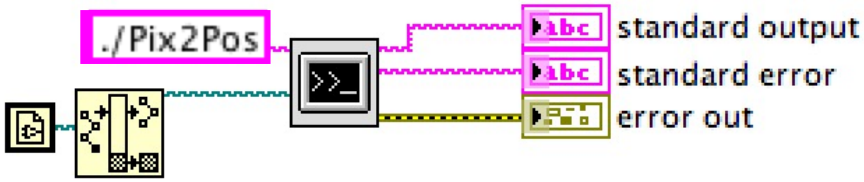
De cette manière il sera possible d'avoir un accès relatif à vos fichiers et exécutable.

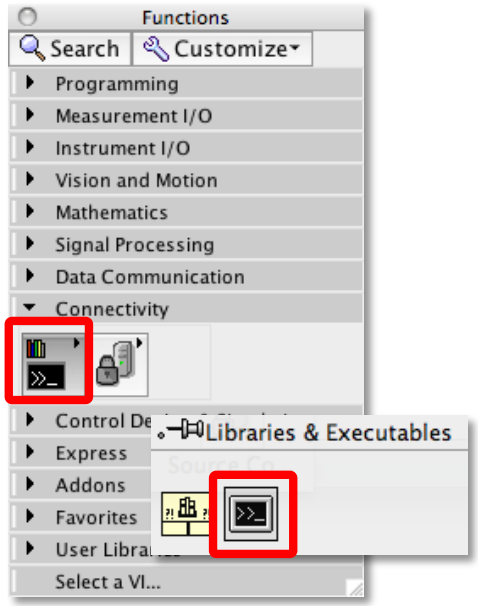
TOUS vos fichiers doivent se trouver dans le même dossier!



Appel d'un programme C++

- **`System exec`** permet d'entrer une commande de la même manière que vous le feriez dans le terminal.
- La ligne de commande pour executer votre programme c++ commence par:
  ./ sous OSX/linux
  cmd /c sous windows
- Cette commande est éxécutée dans le dossier courant (si cela a un sens)
- Il est possible de recupérer le contenu de 'cout' et 'cerr'
- Les erreurs d'execussions sont retournées dans error, Attention cerr ≠ error
- Référez vous à l'aide pour les options supplémentaires



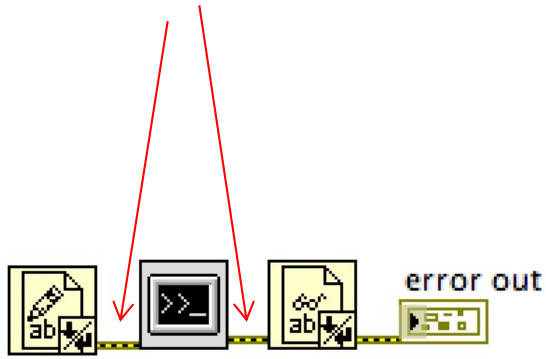


Ordre d'exécussion

Assurez vous de l'ordre d'exécution des Vis (écriture de pixmap.bin, appel du programme C++, lecture des coordonnées) en connectant le cluster d'erreur entre les Vis.

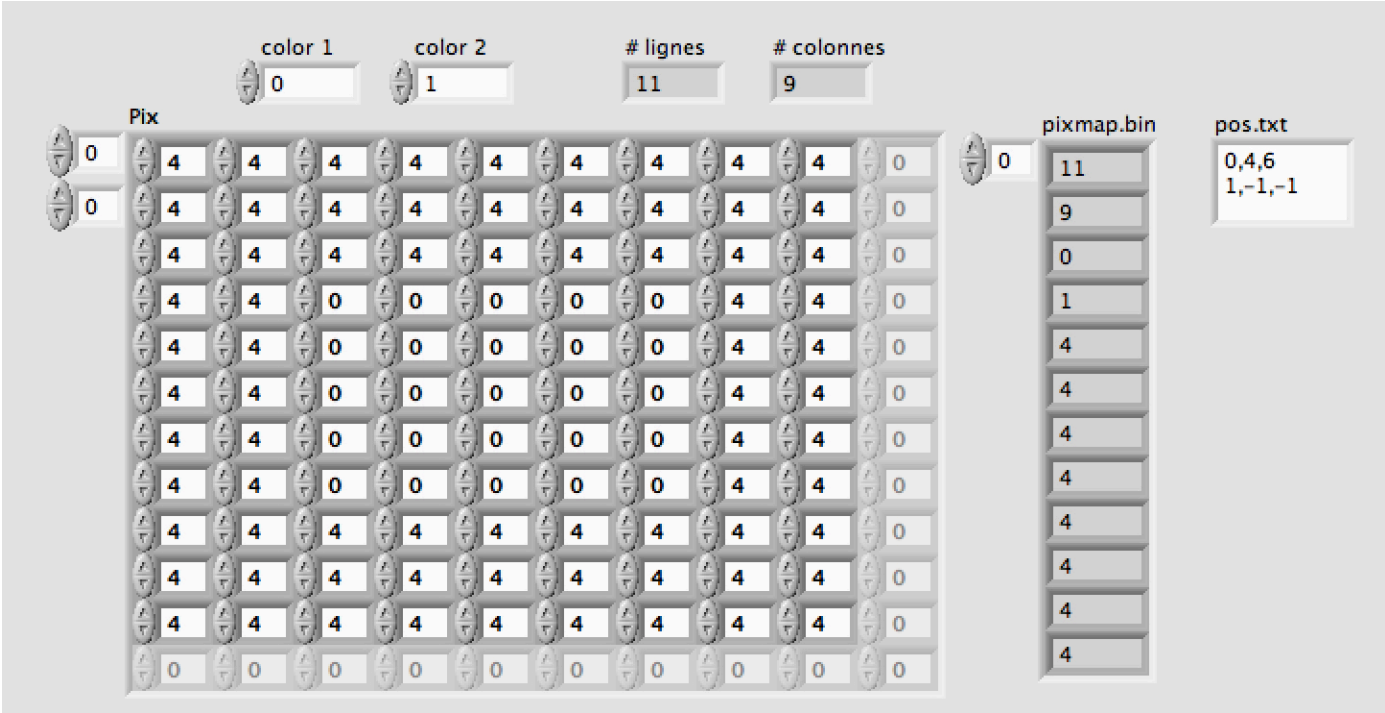
Ne oublier d'afficher l'erreur.

Nous reviendrons plus en détails sur le séquencement des opérations la semaine prochaine.



Test final

- Ajoutez une champs supplémentaire pour afficher le contenu de **pos.txt**
- Assemblez les différentes parties (creation pixmap + écriture de pixmap.bin + appel exe + lecture de pos.txt) pour avoir le programme complet
- Testez votre programme LabVIEW avec des valeurs valides puis avec des valeurs invalides, ex. couleur pattern autre que 0,1 ou pattern 4 x 4, etc.
- Comment réagit votre programme C++ en cas d'erreur ?



A tester :

- Le tableau **pixmap.bin** correspond aux *controls* de votre *front panel*
- L'appel à votre exécutable fonctionne
- Les coordonnées retournées sont correctes
- Les erreurs retournées (ou non) sont correctes
- Que se passe-t-il si vous entrez des valeurs erronées ?

Done!