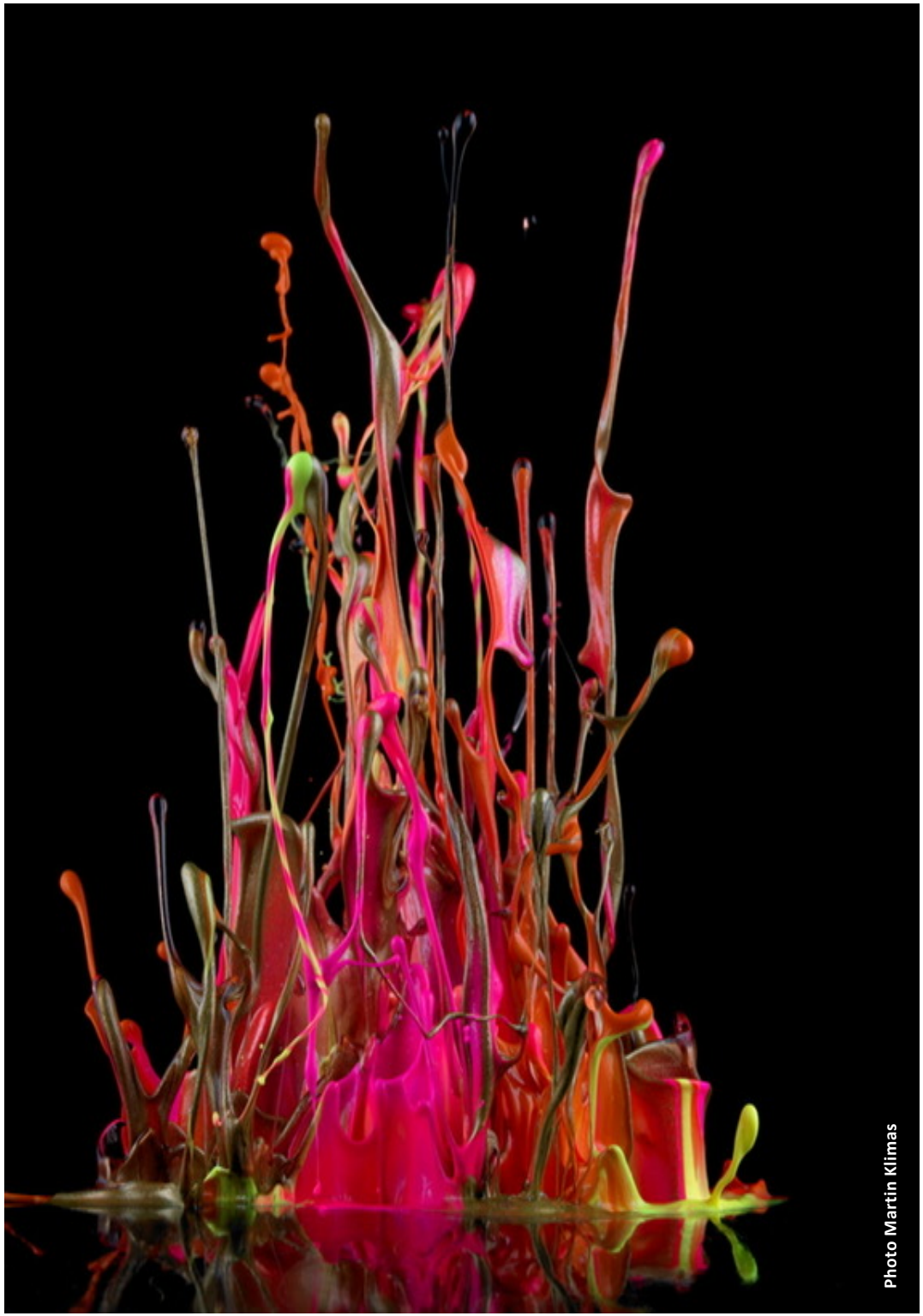



EPFL

# Programmation pour Ingénieur

*Exo. Prj. LabVIEW – Oscillo2data*
*my Third VI*

*ME 3e semestre*

*Rev. 2024.1*

Christophe Salzmann

Laboratoire d'Automatique

# My Third VI

But:

Tester votre programme C depuis LabVIEW

Etapes:

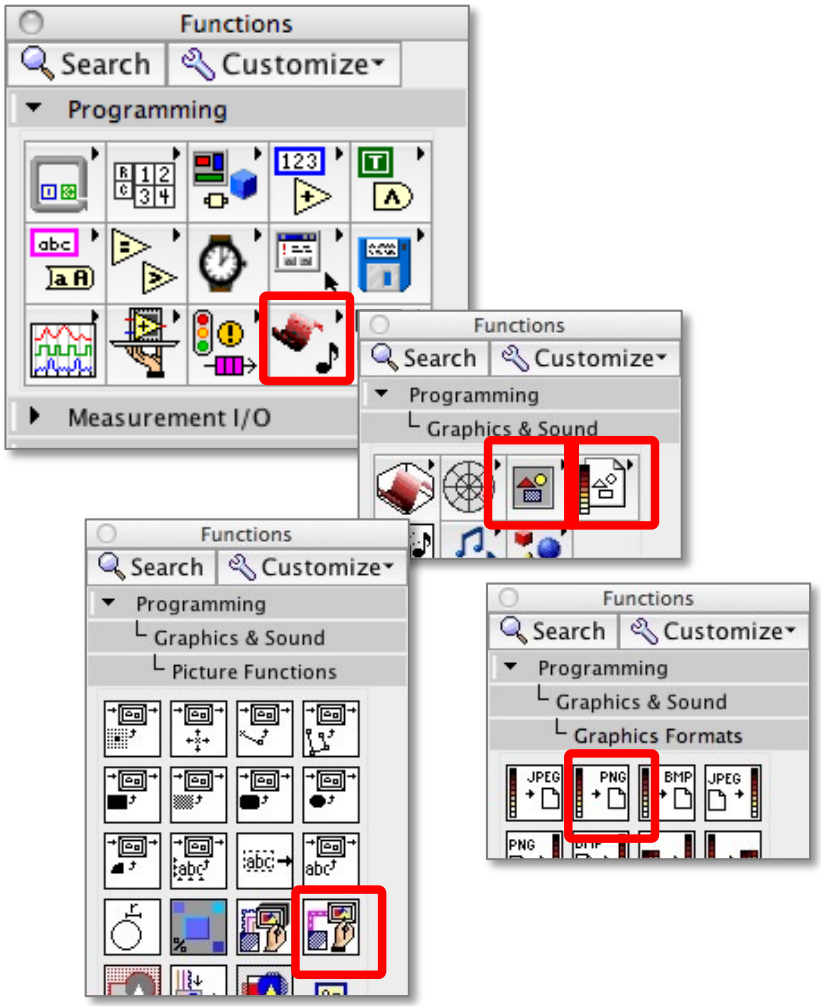
- Lecture du fichier contenant l'image png
- Affichage de l'image
- Accès aux éléments de l'image
- Création du vecteur avec profondeur, largeur, hauteur
- Ecriture du vecteur (P, L, H) dans **Pixmap.bin**
- Ecriture du vecteur **image** dans **Pixmap.bin**
- Appel de votre code C
- Vérification que tout c'est bien passé dans la partie C.

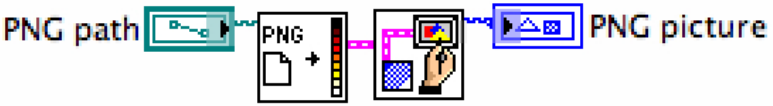
A faire:

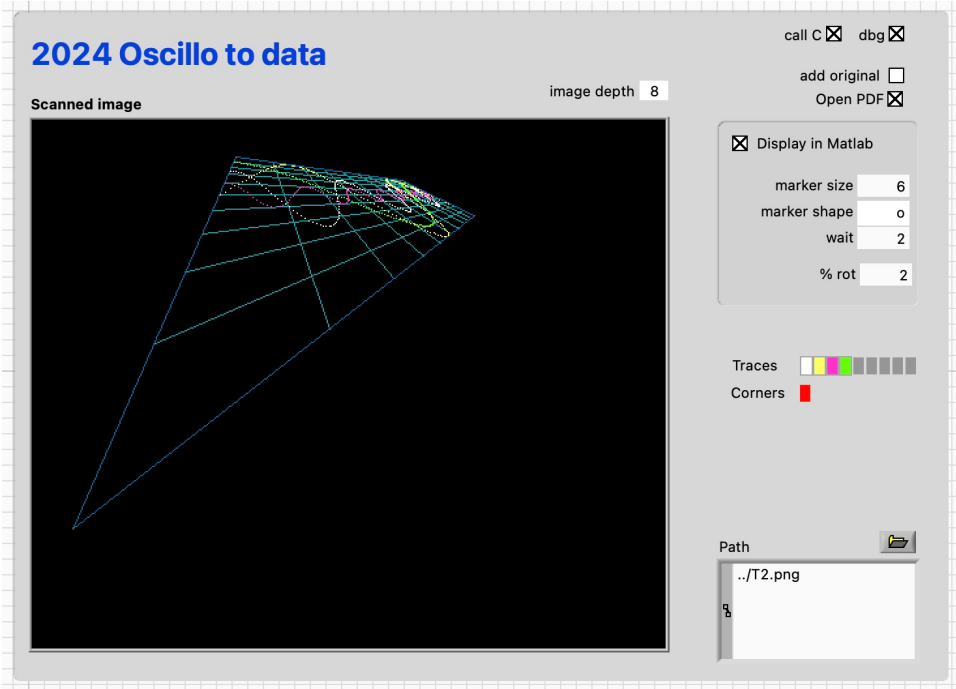
- Lecture du fichier **Traces.txt** et mise en forme
- Sélection des vrais couleurs (non-indexées) des traces et des coins

# Lecture d'une image png

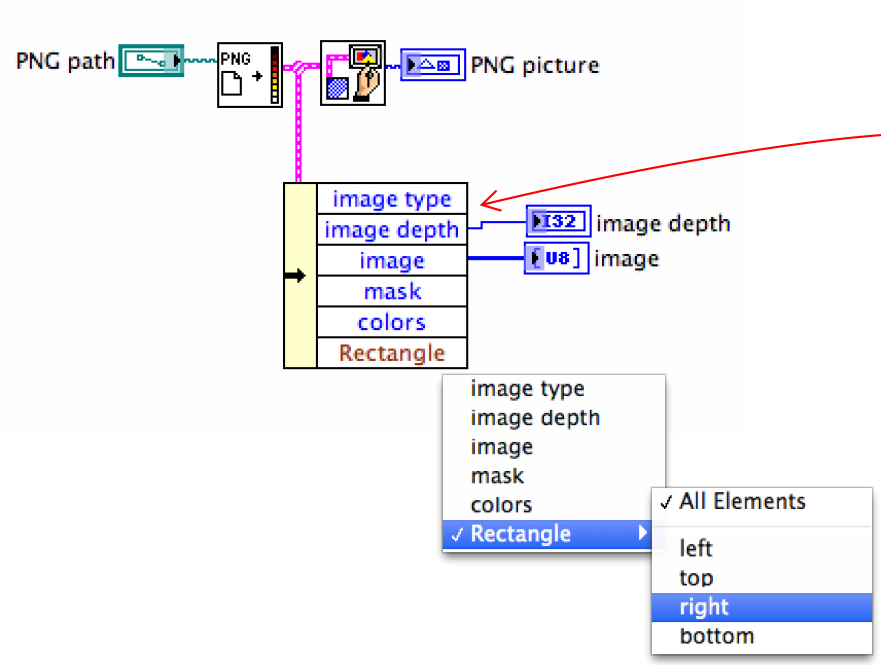
## En seulement 2 Vis!!

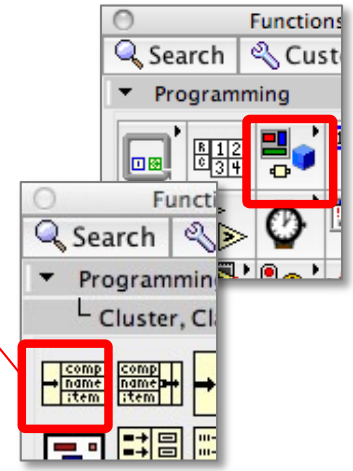


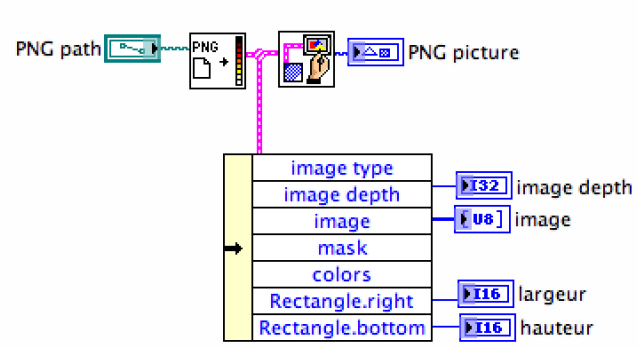



# Accès aux données de l'image



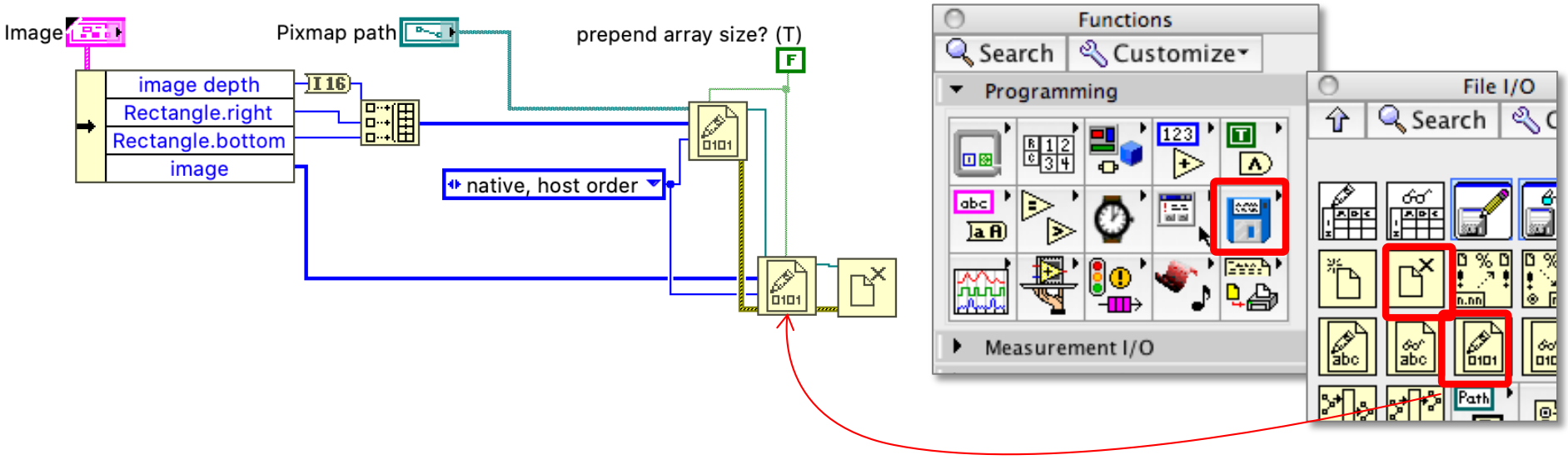




- Connectez **UnBundle By Name** pour accéder au donnée de l'image.
- Dès que vous connectez le fil de l'image, vous verrez apparaître les valeurs de celle-ci, au besoin agrandir **UnBundle By Name**
- Faire un clic de droite sur **Rectangle** pour sélectionner les éléments reprenstants la largeur et la hauteur

# Sauvegarde de Pixmap.bin

La sauvegarde se fait en 2 étapes. La première consiste à sauver la profondeur, la largeur et la hauteur de l'image (3 x I16) et la seconde consiste à sauver les pixels (L x H x U8).

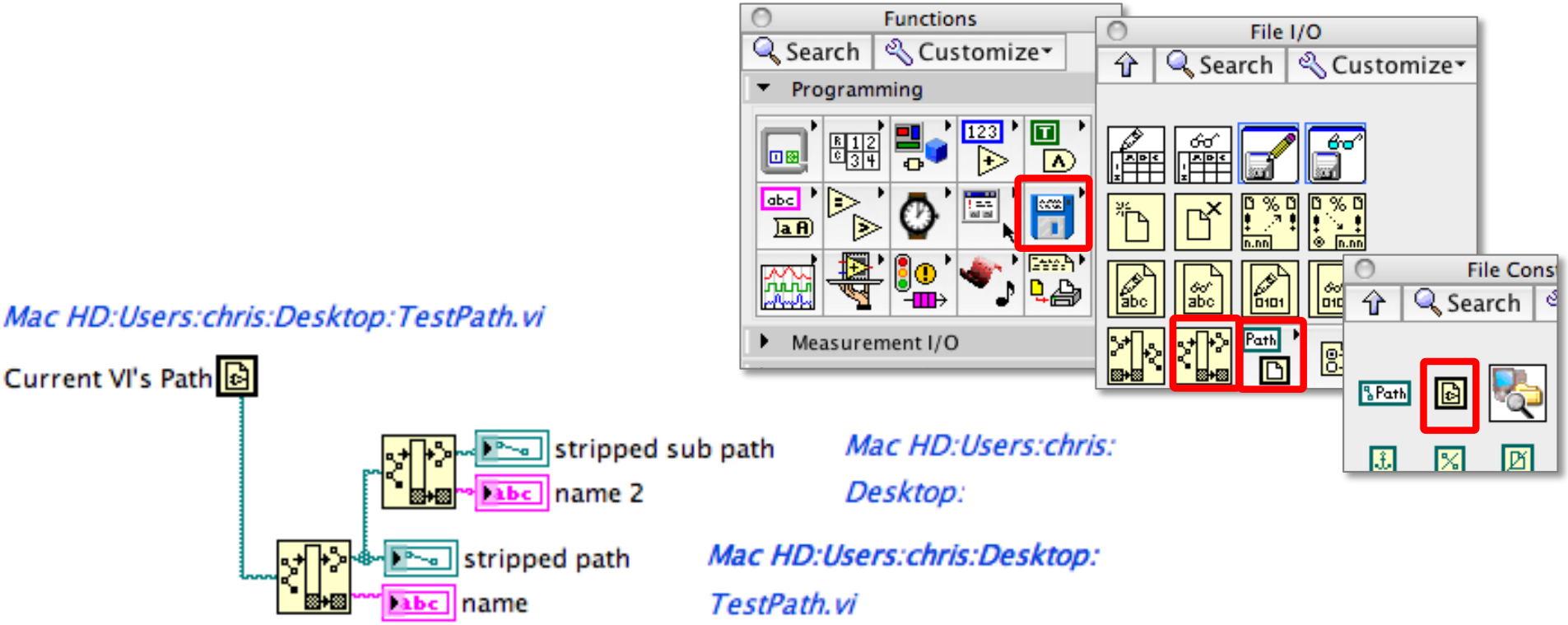
Connectez votre array au VI **Write to Binary file**. Afin de ne pas ajouter la taille du tableau au début du fichier mettez **`prepend array size? (T)`** à faux (F). De même vous devez ajuster *l'endianess* des données sauvée sur votre disque dur (**`little-endian`** ou **`native, host order`**).




Vous devez ajouter une 2e fois le VI **Write to Binary file** pour cette fois écrire le contenu d **image**, une fois terminer vous devez fermer le fichier (**Close File**). Sur cette capture d'écran, le fil d'erreur n'est que partiel, à vous de l'intégrer correctement.

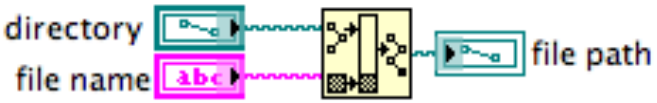
# Chemin courant

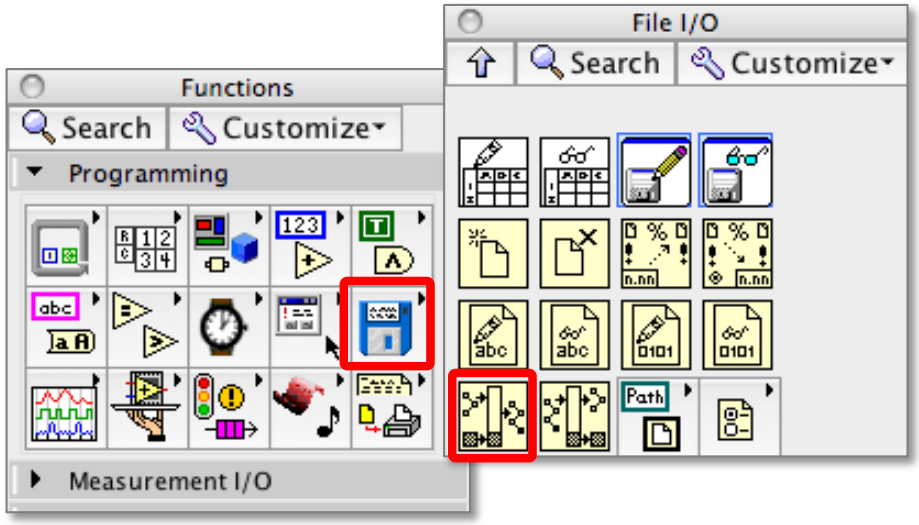
- La constante **`Current VI's path`** donne le chemin complet sur le VI courant (i.e. le chemin de celui qui contient la constante).
- De cette manière il sera possible d'avoir un accès relatif à vos fichiers et exécutable.

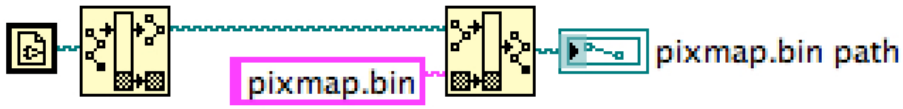


# Chemin courant

- La fonction **`Build path`** permet de créer un chemin complet sur un fichier.
- De la même manière il est possible de créer un chemin complet relatif au VI courant, par exemple pour *Pixmap.bin*

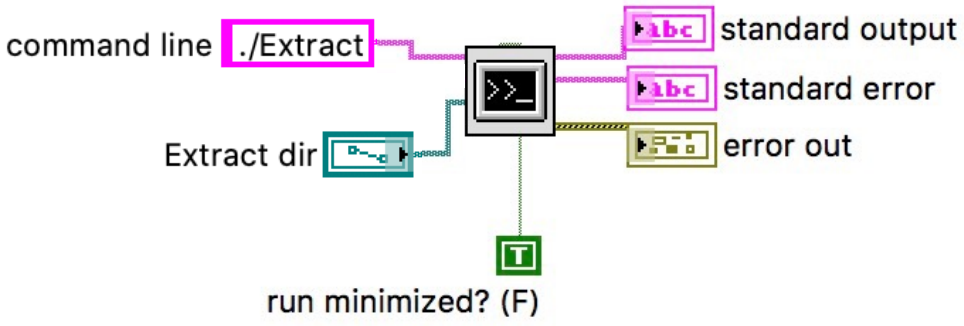


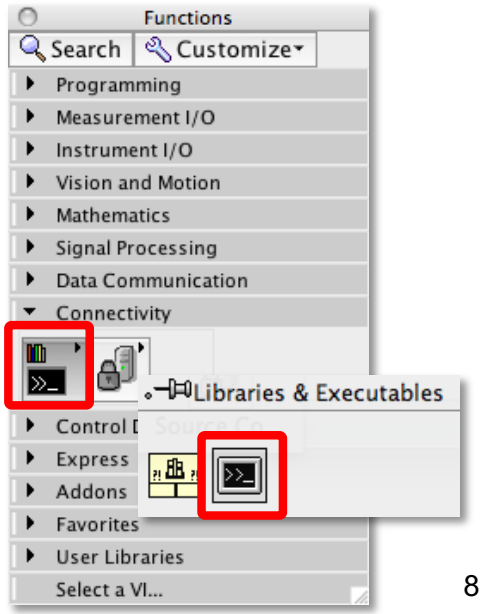



# Appel d'un programme C

- **`System exec`** permet d'exécuter une commande de la même manière que dans le terminal.
- La ligne de commande (**command line**) pour executer votre programme C commence par:
  **./** sous OSX/linux
  **cmd /c** sous windows
- Vous devez ajouter à la **commande line** les chemins de **Pixmap.bin** et **Traces.txt** (non montré dans la copie d'écran ci-dessus)
- Cette commande est éxécutée dans le dossier indiqué (**Extract dir**)
- Il est possible de récupérer le contenu de 'stdout' et 'stderr'
- Les erreurs d'execussions sont à retourner dans le cluster **error out**, à vous de le construire.
  Attention **stderr** ≠ cluster **error out** !

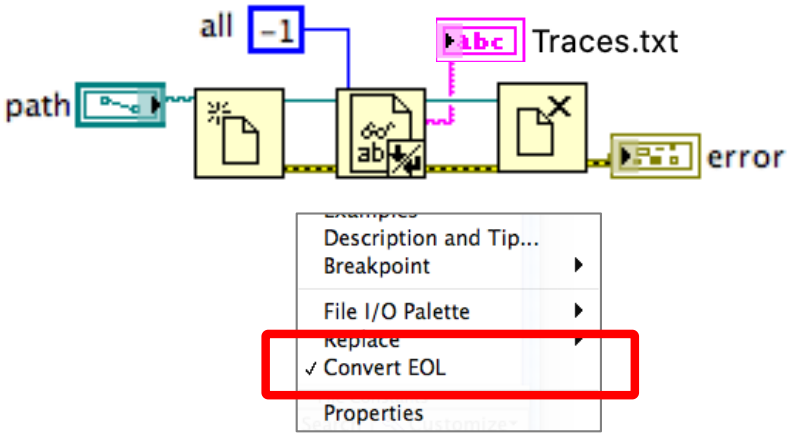


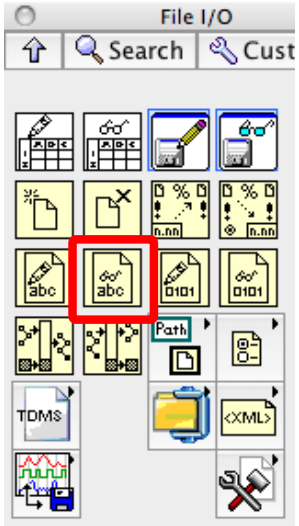




- Après l'appel au C, **stdout** contiendra les indexes des couleurs des traces et des coins
- Référez vous à l'aide pour les options supplémentaires

# Lecture du fichier Traces.txt

- Comme en C/C++vous devez ouvrir le fichier, lire les données et fermer le fichier.
- LabVIEW gère les fichiers au format txt (**`read from text file`**) et au format binaire.
- LabVIEW adapte le cartère de fin de ligne (clic de droite, option **Convert EOL**) en fonction de la plateforme sur laquelle est éxécuté le VI.
- Il est possible de lire un nombre défini de caratères dans le fichier, mettre '-1' pour lire l'entier du ficheir



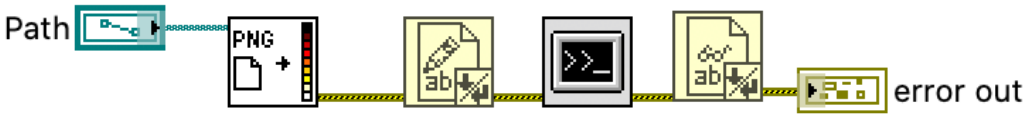

*Regarder l'aide en ligne pour voir sous quelles conditions* ***read from text file*** *peut être employé sans avoir à ouvrir et fermer explicitment le fichier.*

# Ordre d'exécution

Assurez-vous de l'ordre d'exécution des Vis en connectant le cluster d'erreur entre les Vis.

1) Lecture du png
2) Création et écriture du tableau Pixmap.bin
3) Appel du programme C
4) Lecture des Traces.txt

Ne oublier d'afficher l'erreur




*mockup*

# A tester :

- Le vecteur (P,L,H et image) sont bien construits et sauvegardés dans **Pixmap.bin**
- L'appel à votre exécutable fonctionne
- **Pixmap.bin** est bien lu par le C
- Les chemins sont corrects
- Les **Coins** et **Traces** retournés sont corrects
- Les erreurs retournées (ou non) sont correctes
- Que se passe-t-il si vous entrez des valeurs erronées ?

# A Faire :

- Regarder le PDF - *Color lookup table (CLUT)* pour trouver les vraies couleurs correspondant aux index des traces et des coins

**Done!**